AFRL-RI-RS-TR-2016-020

A SCALABLE, OPEN SOURCE PLATFORM FOR DATA PROCESSING, ARCHIVING AND DISSEMINATION

MDA INFORMATION SYSTEMS LLC

JANUARY 2016

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2016-020 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /
CRAIG ANKEN
Work Unit Manager

/ S /
MICHAEL J. WESSING
Deputy Chief, Information Intelligence
Systems and Analysis Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JANUARY 2016	FINAL TECHNICAL REPORT	Nov 2012 – Nov 2014

4. TITLE AND SUBTITLE

A SCALABLE, OPEN SOURCE PLATFORM FOR DATA PROCESSING, ARCHIVING AND DISSEMINATION

5a. CONTRACT NUMBER
FA8750-13-C-0016

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62702E

6. AUTHOR(S)

Samuel Park, Chris Mattmann, Yolanda Gill

5d. PROJECT NUMBER
EF00

5e. TASK NUMBER
81

5f. WORK UNIT NUMBER
69

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
MDA Information Systems LLC
820 West Diamond Ave., Suite 300
Gaithersburg, MD 20878

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RIEA
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2016-020

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Building on the Object Oriented Data Technology (OODT) big data toolkit developed by NASA and the Work-flow INstance Generation and Selection (WINGS) scientific work-flow system developed by the Information Sciences Institute at the University of Southern California, we focused analytics and work-flows appropriate to several challenge big data problems and demonstrated the utility of OODT-WINGS in addressing them. Specific demonstrated analyses address i) crawling and indexing the XNET contents in a quick-reaction demonstration in January 2013, ii)a "data triage" characterization of the Kiva loan system, iii)contributory work done on 5 of 6 XDATA summer 2013 workshop challenge problems, and iv)demonstration of end-to-end processing work-flow in conjunction with the USC Institute for Creative Technologies end-user interface at the September 2013 DARPA open house. Two papers published during this time addressed i)the overall "data triage" use case, and II) work flow characterization during time-bounded operations.

15. SUBJECT TERMS
Open source software, Apache, Object Oriented Data Technology, OODT, semantic work-flows, WINGS, big data, work-flow management

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**CRAIG ANKEN**
U	U	U	UU	45	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

Table of Contents

List of Figures

1.0 EXECUTIVE SUMMARY

MDA Information Systems LLC (MDA) assembled a team of open source industry leaders and provided technical software development and demonstration of the software's capability to the DARPA XDATA program and various DARPA stakeholders and potential customers. NASA's Jet Propulsion Laboratory (JPL) developed Object Oriented Data Technology (OODT) for a decade before transitioning it to the Apache Software Foundation (ASF) in 2010. The Information Sciences Institute at the University of Southern California (USC/ISI) developed the Workflow Instance Generation and Selection (WINGS) over a comparable period. Both were used to develop and deliver capability to the XDATA software toolkit.

After the first "PI Mini-meetings" held in December 2012, a month of defining a development path complementary to other performers' offerings was capped off by a special quick response drill in late January 2013 that resulted in fully operational software that showed the results of crawling the XDATA Network (XNET) contents in a fully searchable form, with both web-site GUI and a demonstration on an iPhone. This was performed only one day after the XDATA kickoff meeting, to highlight the responsiveness of the approach.

The actual scope of our development and demonstration was focused during review and feedback received during the DARPA site visit to JPL in April 2013. The guidance received was to focus on the adaptive OODT-oriented use cases, especially as addressed an "unknown dataset" case that we called the "Zero Dark Thirty" scenario. We agreed to illustrate this capability by using the large touchscreen interface being developed by another XDATA performer team, from the Institute for Creative Technologies (ICT) at the University of Southern California.

The MDA-JPL-ISI team conducted several illustrative demonstrations during the one-year time-period of execution, including 1) Quick-turn XNET search (January 2013), 2) Support of the ICT end-user interface with geographically-selectable Twitter data during the first week of the 2013 summer workshop, 3) the first demo and mid-point review conducted during the summer workshop:

- **1st Demo (6/14/2013) with Live REST endpoints**
  - Demo on the first Friday of the workshop
  - Showing actual ingested and searchable data accessible via simple link
- **1st Mid-Point Review (6/27/2013)**
  - Initial Survey illustrating Data Triage and Workflow Execution Monitoring
  - Characterization of Kiva and Twitter data sets
  - Initial run metrics captured via OODT OpsUI monitoring

Publications included one that illustrated the Zero Dark Thirty use case for a general reader and another more technical study of characterizing and executing workflows in time-bound scenarios.

2.0 INTRODUCTION

We integrated our offerings into the DARPA XDATA software toolkit by leveraging and building on the following team member contributions, all of which are recognized open source Apache technologies.

- Apache OODT, an information integration and science data processing framework
- Apache Tika, a MIME-type identification, classification, and parsing/extraction framework
- WINGS workflow system, pioneered by USC ISI

Our team emphasized the use of the Apache (OODT) set of tools used for quickly adapting existing and developing code to data in a flexible opportunistic way combined with the disciplined semantic

workflow technology embodied in (WINGS) software. Apache™ OODT has been central to the work done for XDATA, comprising the main technical approach taken during the 2013 summer workshop. WINGS has been a key adjunct, and has provided a natural development path for elaborating and specializing workflows for application to the XDATA challenge problem sets.

While open source project integrations regularly redevelop the "glue code" software that connects the projects, this ad hoc approach is often hard to scale and maintain. We addressed this by developing an analytic and extract-transform-load workflow specification (WINGS), information and metadata extraction within those workflows (Tika, OODT), data/metadata cataloging/archiving (OODT), algorithm integration (OODT) and reliable, distributed and replicated data storage.

The integration of massive datasets of disparate form and structure required development of equally diverse algorithms and tools to achieve integrated system functionality. Facing the challenges of software integration of diverse products required a principled approach to encapsulation and wrapping of software, exactly the key capabilities of OODT and Wings. Integrating these previously disparate technologies brings together several open source communities.

3.0 TECHNICAL DEVELOPMENT (METHODS, ASSUMPTIONS, AND PROCEDURES)

We delivered the DARPA XDATA required software integration using open source Apache™ technologies in a management framework that leverages continuing presence in the XDATA Facility, facilitating frequent on-site coordination with DARPA and other on-site personnel.

MDA coordinated and managed the overall effort. The team used Apache™ OODT as the information integration and science data processing framework, Apache™ Tika for identification, classification, and parsing/extraction. These were enabled with automated workflows from the WINGS system.

3.1 TECHNICAL APPROACH

Our approach made minimal assumptions about the underlying implementation platform, programming language, etc. of the algorithm or visualization integrated, using the Apache™ OODT Catalog and Archive System (CAS) Product Generation Engine (PGE) "wrapper" technology. Using these, we specified:

- Input Files/Metadata (provided by the OODT File Manager)
- Required command line flags, environment variable settings, or input settings
- How to execute the algorithm or visualization (e.g., shell script, Python commands, or other)
- How to extract metadata (leveraging Apache™ Tika) from the generated output data files/streams
- How to catalog/archive and make available all output for subsequent algorithm or visualization executions on top of the substrate

The team's approach with software development and system integration resulted from 10 years of experience and 100s of FTEs using and integrating disparate technologies, in order to build a cohesive final solution. Experience gained from organizing and maintaining projects using ASF methodologies is paramount when working to merge several technologies in to a single solution. The "Apache™ way" means that there must be 100% transparency with the development efforts whereby all stakeholders can interact with each other in a way that the rest of the community (members involved) can accurately assess and take action.

The ASF is unique in that members are not restricted by location or to a specific outside organization. Project team members can be physically located across the globe which makes project transparency that much more important when ensuring project success.

3.2 DEMONSTRATION OF UTILITY FOR POTENTIAL CUSTOMERS

Part of this effort has been to provide an easily understood visual demonstration of the "big data" processing operation conceived as a continually operating system. This was demonstrated at the DARPA I2O Open House with the aid of a poster (Figure 1) that visually depicted the connection of our technology (middle "Data Pipeline" in the Figure) to the overall DARPA XDATA portfolio of performers (the top center "XDATA Workflow" in the Figure).

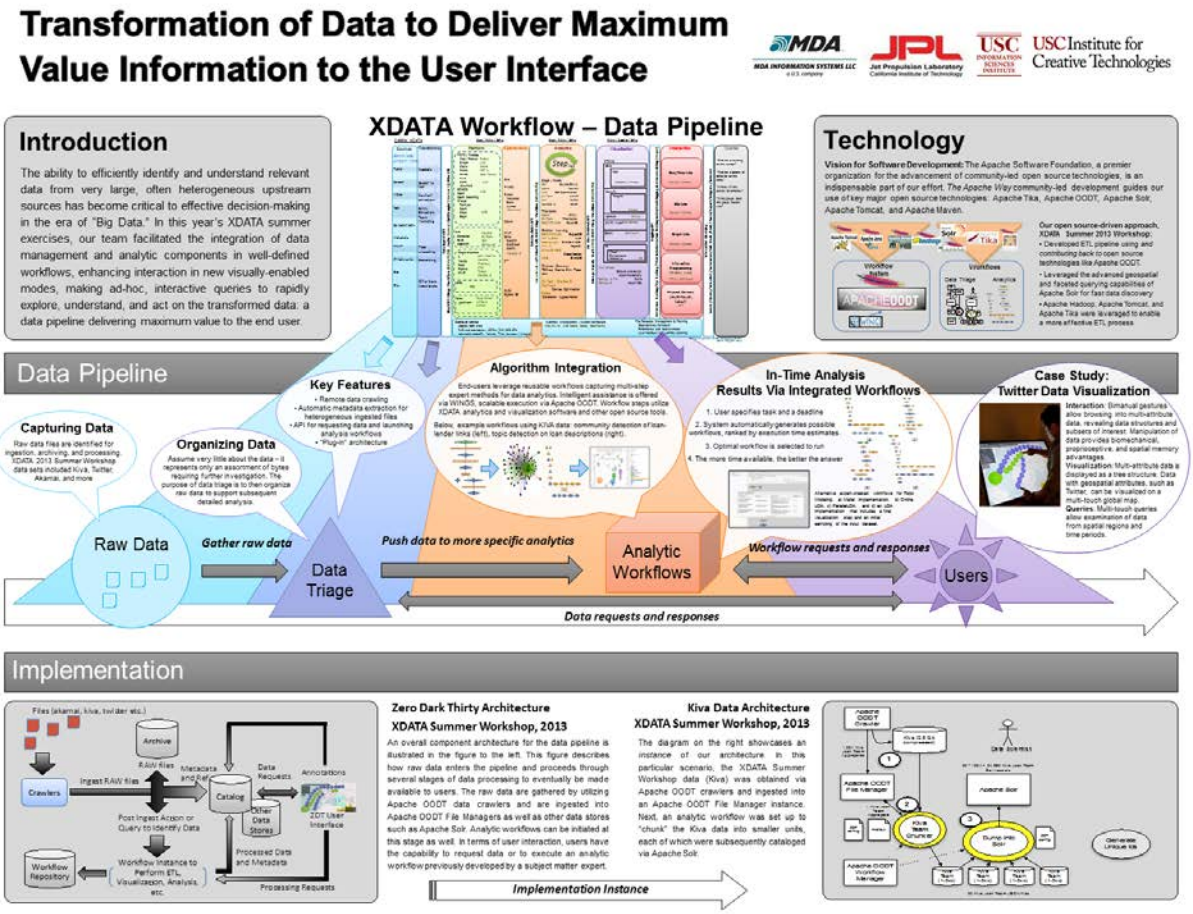


Figure 1: Transformation of Data Demonstrated in a High-Level Workflow

3.3 SPECIFIC TECHNICAL IMPACT AND IMPLEMENTATION

The software development has been focused on a specific "data triage" use case that we sometimes refer to as the "Zero Dark Thirty" use case. The prospective impact from a customer perspective was developed in a paper published in *Geospatial Intelligence Forum* (July-August 2013 issue, Vol.11, #5, pages 25-26), as shown in Appendix A of this report.

The typical use of this "Zero Dark Thirty Architecture" as a template for particular application is illustrated in the bottom of Figure 1. An overall component architecture for the data pipeline is illustrated here at bottom left, describing how raw data enters the pipeline and proceeds through several stages of data processing to eventually be made available to users. The raw data are gathered by utilizing

Apache OODT data crawlers (akin to the Apache™ Nutch crawling system) and are ingested into Apache OODT File Managers as well as other data stores such as Apache™ Solr. Analytic workflows can be initiated at this stage as well. In terms of user interaction, users have the capability to request data or to execute an analytic workflow previously developed by a subject matter expert. Once the data is made available in Solr it is ripe for analysis in terms of distribution values; faceting, range queries, fielded queries, and other analytics.

The particular implementation instance is derived from this architecture for a particular application. The one shown in Figure 1 (bottom right) is the application to the Kiva data analysis challenge. In this particular scenario, the XDATA Summer Workshop data (Kiva) was obtained via Apache™ OODT crawlers and ingested into an Apache™ OODT File Manager instance. Next, an analytic workflow was set up to "chunk" the Kiva data into smaller units, each of which were subsequently cataloged via Apache Solr.

We did a "quick-turn" exercise in January 2013 to illustrate the flexible adaptive nature of Apache™ OODT and related capability. This exercise crawled the XNET and did an initial characterization of it. As a side benefit it helped our team understand our technical offering and how it related to the offerings of other XDATA performers. Figure 2 shows an example of the GUI for the XNET Search capability

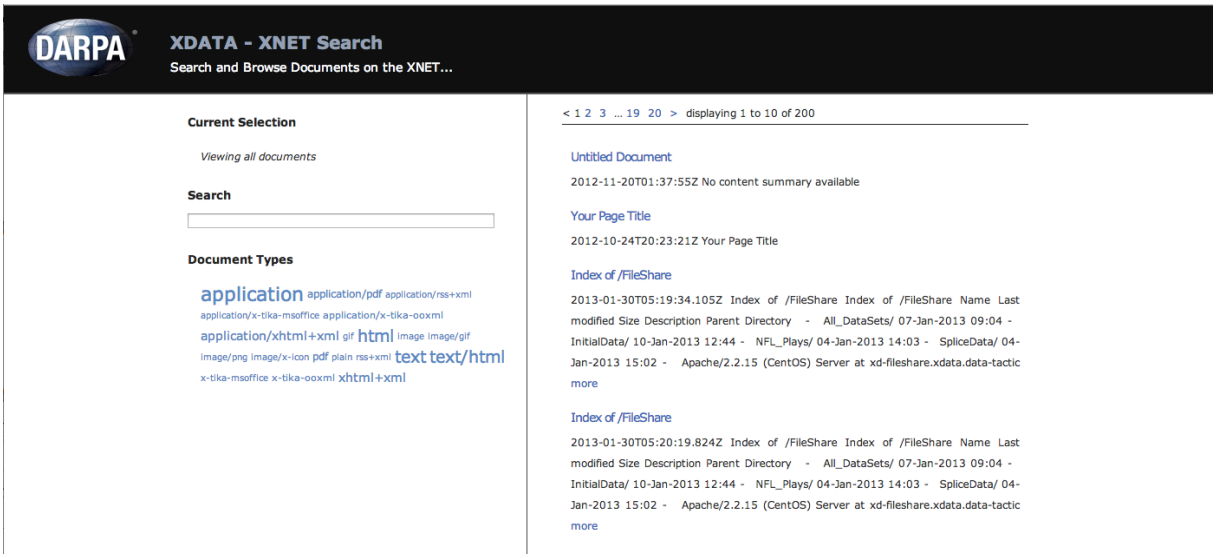


Figure 2: Graphical User Interface (GUI) Display of the XNET Search results – developed during an illustrative coding sprint in late January 2013 -

The instantiated software configuration used to do the XNET Crawl is shown in Figure 3.

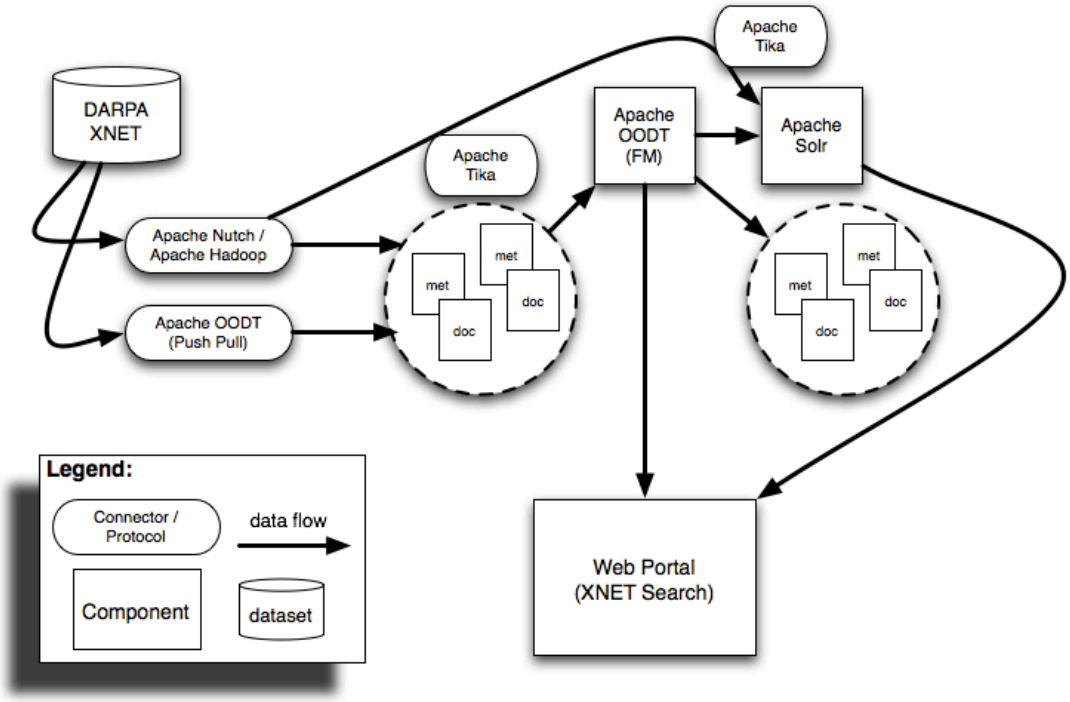


Figure 3: Software Configuration for XNET Crawl

Illustrative metrics derived from the XNET Crawl exercise are shown in Figure 4. These metrics demonstrate that our search algorithm and prototype had to deal with more flexible content types, more information downloaded, and ultimately scaled in near-linear time to handle the data.

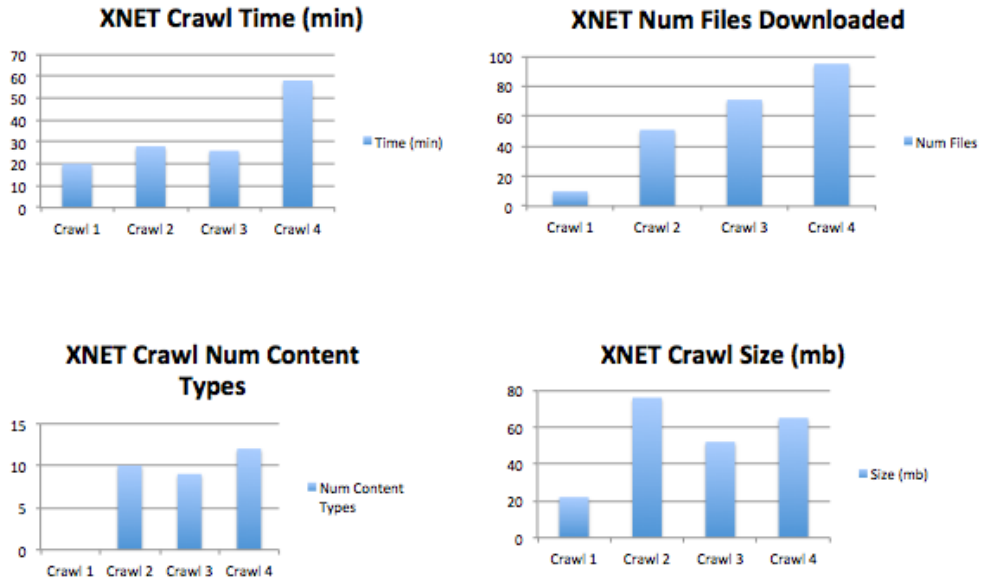


Figure 4: Performance metrics for the January 2013 XNET Crawl Quick-Turn Exercise

Guidance received from DARPA at the 25 April 2013 site visit to JPL was to

1. Re-orient our current effort to support this niche “data triage” capability.

2. Develop a data triage scenario
3. Illustrate this scenario by working with XDATA Visualization Performers, particularly USC's Institute for Creative Technology (credited in Figure 1, for instance).

Our team took a lead role in the XDATA API Working Group, providing concrete examples of APIs that were put on the XDATA wiki (http://lists.data-tactics-corp.com/pipermail/api-xdata/2013-February/000070.html), developing the XDATA component classification (jointly with M. Massie of the Berkeley AMPlab), and developing a logging API (jointly with C. Forlines of Draper Laboratory). Specific APIs that we produced, relevant to our work and XDATA, were:

- Process Control (OODT/Hadoop)
    - Health Monitoring (OODT)
    - Provenance/Tracking (OODT/Wings)
    - File/Metadata Listing (OODT)
- Data Curation (Tika/OODT)
    - Dynamic and Exploratory Metadata Extraction
    - File Staging/Ingest from Staging area to repo
    - Dataset/Policies
    - Dataset Collections and Metadata Management
- Data Access (OODT/Solr)
    - Datacasting/RSS
    - RDF data representation
    - Download/subset (as zip along with metadata).

In addition, we pursued integration of the WINGS workflow system with Apache™ OODT. The initial combined WINGS-OODT Integration (called "WOOT") is shown in Figure 5.

The goals for this integrated development were:

- Facilitate rapid integration of analytics and visualization tools
    - from open source projects
    - from other XDATA performers
- Focus on core contributions
    - OODT and related Apache software products
        - Central initial data discovery, handling and disposition
    - Demonstration of key big data capabilities
        - E.g., Tika automated file detection and metadata extraction at scale
    - Basic capability for non-experts to use complex workflows with WINGS

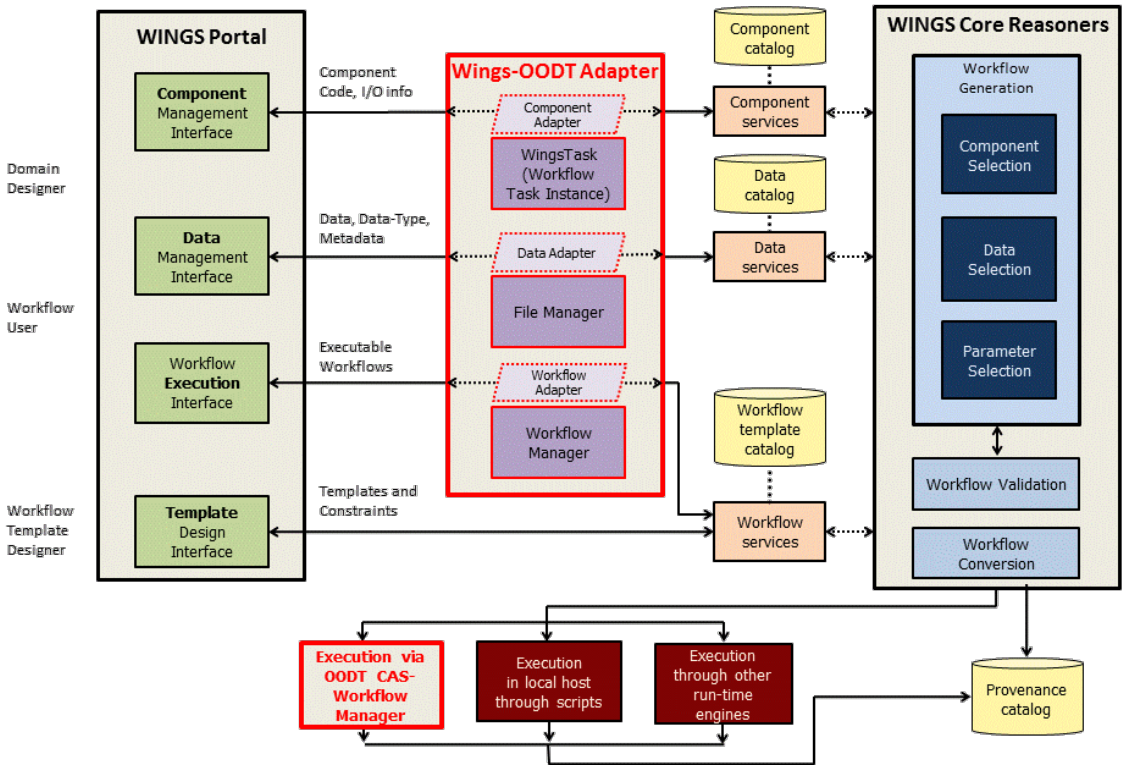


Figure 5: Initial WINGS-OODT Integration (March 2013)

4.0 EXECUTION PLAN

The tools that were developed for XDATA were integrated in the same way as the core Apache™ projects from which they are derived. The key differentiator is that the supporting community is DARPA and associated stakeholders.

4.1 SOFTWARE INTEGRATION

The platform architecture is an integration framework that provides mechanisms for easily receiving algorithms and visualizations and for encapsulating these algorithms and visualizations as software components in the OODT and WINGS infrastructures (see Workflow Compilation and the OODT Product Generation Executive or PGE wrapper in Figure 6). The system operator (human or automated software) interacts with the platform via the upper-most layer and its user-facing RESTful services, RSS feeds, and/or the XML-RPC protocol, once the algorithms and visualizations have been integrated. WINGS and the Apache™ OODT PCS Operator Interface provide user interfaces for managing, and monitoring job submissions, displaying visualizations. Behind the scenes, WINGS and OODT provide toolkits, software APIs and workflow specification languages to describe algorithms, and visualizations, and to model their data flow, and control flow.

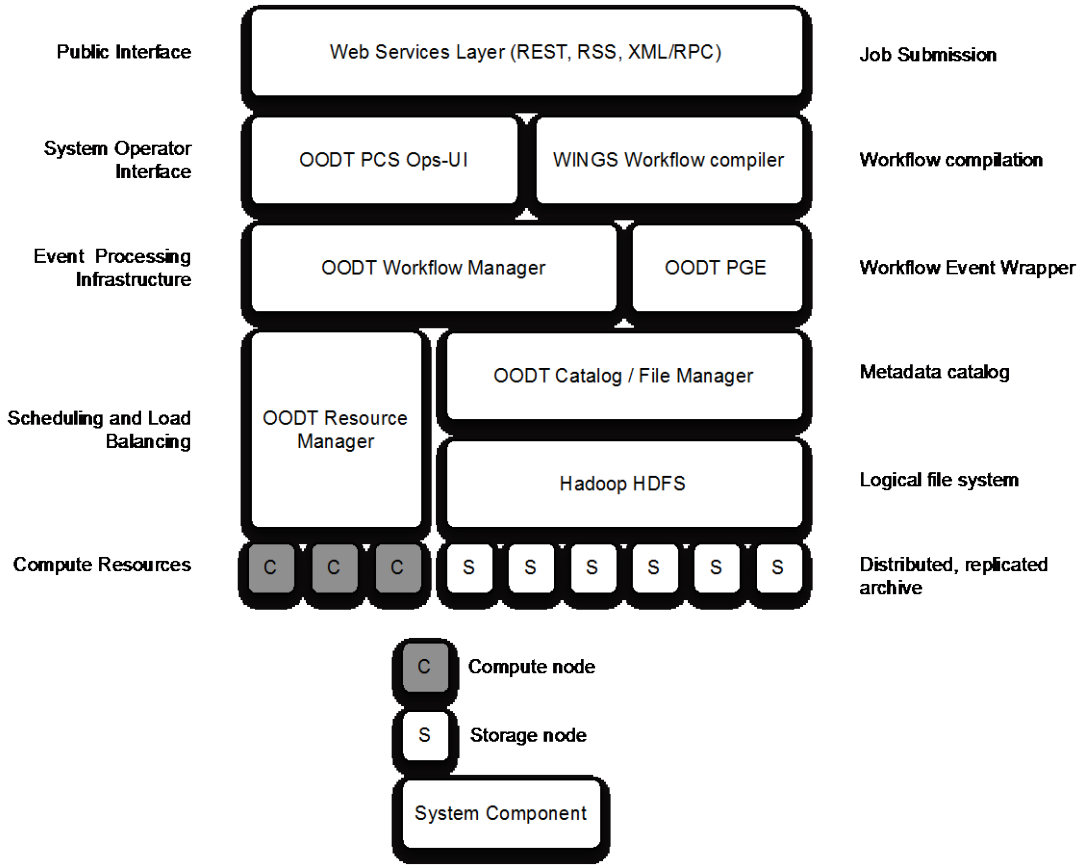


Figure 6: Architecture "To-Be" Goal for XDATA Incorporation of OODT and WINGS

Our system provided file and metadata management (including content detection and analysis) using the Apache™ OODT file management component (as shown in Figure 1) and then stored those files in the host file system (Hadoop HDFS being shown as an example). File metadata, classification and analysis will be provided by the Apache™ Tika framework. Ultimately as demonstrated in Figure 6, the system realized our original intention for providing an analytic and integration platform for XDATA software.

5.0 TECHNICAL ACCOMPLISHMENTS (RESULTS AND DISCUSSION)

Beyond the illustrations of capability demonstrated early in the project, technical accomplishments during this project execution were i) analyses of the challenge problems during the 2013 Summer Workshop (Section 5.1) and ii) highlights of participation in the on-site Summer Workshop collaborative analyses (Section 5.2.

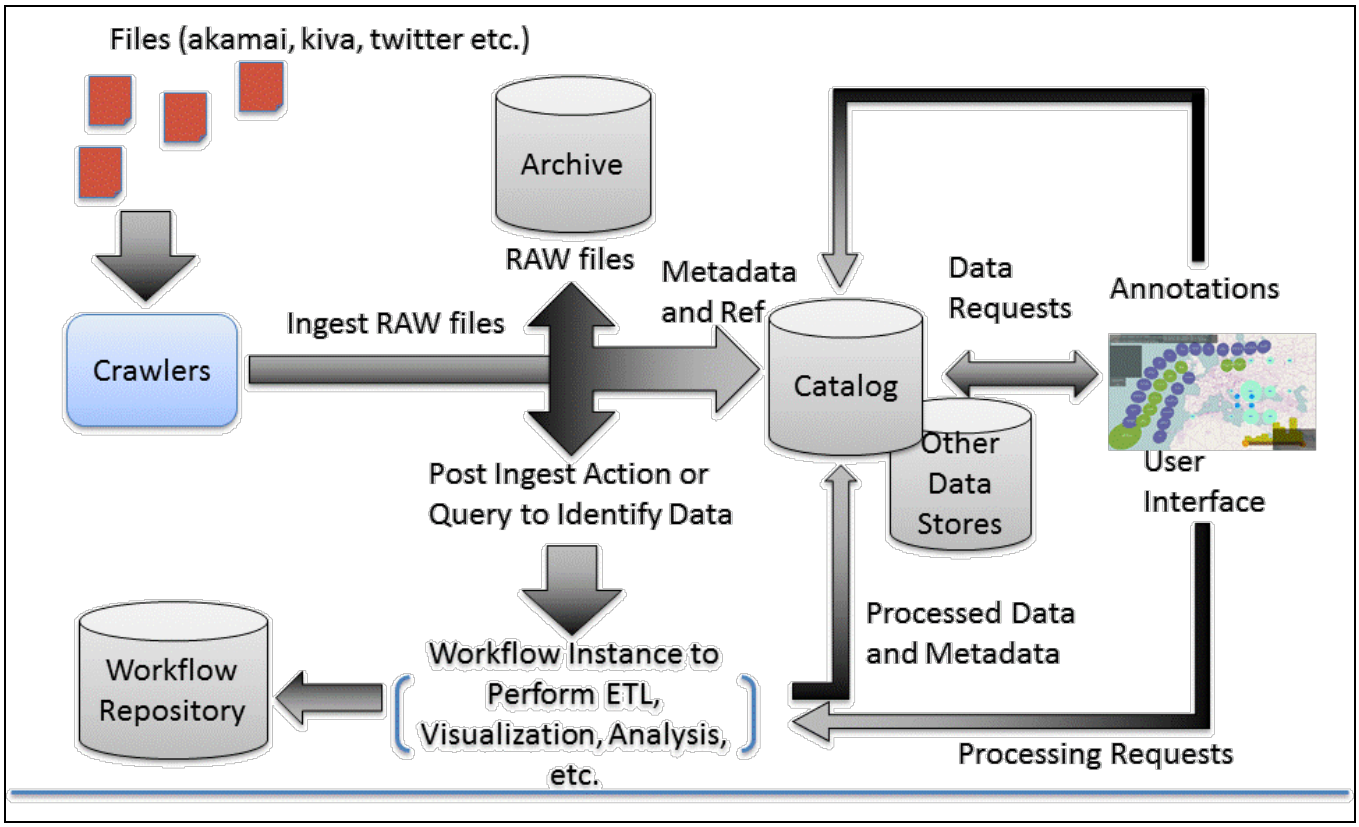


Figure 7: Approach for Summer 2013 Workshop Analyses

5.1 SUMMER WORKSHOP 2013

There were six challenge problems presented to the XDATA performers during the 2013 Summer Workshop. The MDA-JPL-ISI team contributed to analyses of 5 of these 6 challenges (BitCoin was not addressed). The analysis is shown in detail for Kiva (Section 5.1.1) and in overview fashion for the other problems, conducted in support of other performers (Section 5.1.2). In addition, we (working with Data Tactics) characterized and compared the general Extract-Transform-Load processes for the challenge problems, the primary results of which are given in Section 5.1.3.

The framework for the analyses is shown in Figure 7. In general, we went back and used this "Zero Dark Thirty" architecture, which made minimal assumptions about the data going in. This effectively regarded the raw data as an undifferentiated "blob of bytes" to investigate, then proceeded from this raw data to various derived products. We allowed annotations (metadata) throughout, derived various from the system (operating, file), characterization by algorithms, and as derived by effort of individual analysts. We captured the linkages between the data internally and externally, handling both structured and unstructured data. The process of producing the data is important, so we captured it, ensuring repeatability and providing a means of tracking and representing provenance of the result of any given analysis.

5.1.1 Kiva Analysis

Kiva is an online collaborative system of providing loans: essentially its own banking system focused on providing capital (from loan providers) to individuals or organizations in need of it (to loan recipients).

We implemented the Zero Dark Thirty architecture to address the Kiva data set, as illustrated in Figure 8. (A high-level depiction of the unified process appears in the bottom of Figure 1.)

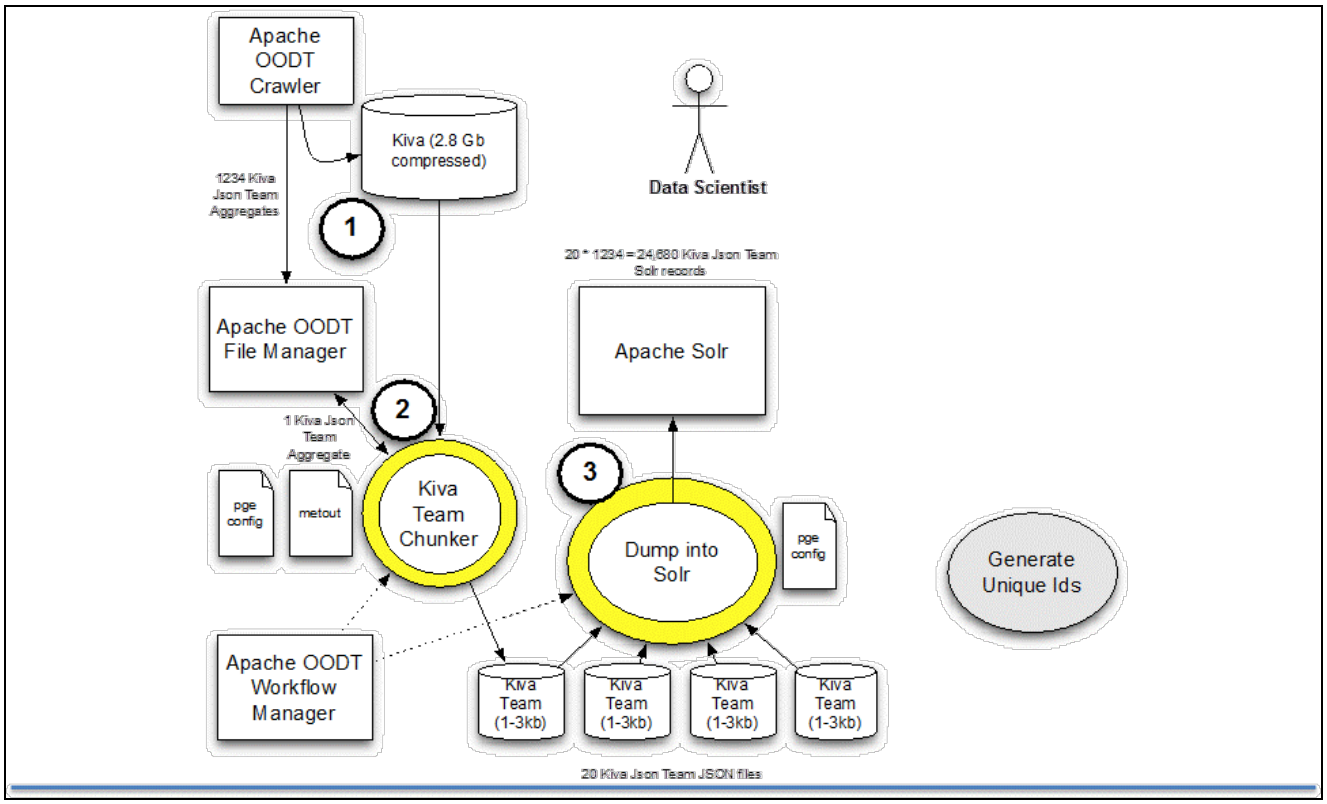


Figure 8: Kiva Analysis Implementation for Summer 2013 Workshop Analyses

The specific steps called out are 1) get the Kiva team aggregates (JSON data) into Apache OODT File Manager for initial handling, 2) characterize data and chunk by size, and 3) Index the chunked data and dump into the Apache Solr database. The end-result is readily available fully indexed and searchable data, easily analyzed via distribution means, different values, and plotting/viz as shown on the next page of the report. This Solr database may now respond to complex ("faceted") queries, as illustrated in the characterization overview shown in Figure 9.

# Loans:	524,514
# Lenders (kiva users):	1,092,480
# Partners:	238
# Transactions:	4,069,217
# Total $ Loaned:	$418M (USD)
# Journal Entries:	307,831

Figure 9: Overview Characterization for Kiva Analysis

Figure 10(a) shows workflows that were used to analyze Kiva data, the one on the right for detecting popular uses of loans and the one on the left for detecting communities of lenders. Creating a workflow does not require a lot of effort. Big data experts can directly reuse the codes that they have.

For detecting popular uses of loans we used expert-created workflows for modeling salient topics in a collection of documents. They all use the popular LDA (Latent Dirichlet Allocation) algorithm to detect

topics. Each of the workflows uses a strategy that works better with large datasets for different reasons: a) efficient LDA through hyperparameter optimization, b) online LDA for incremental processing of streaming data, c) a parallel LDA implementation, and d) sampling to reduce the size of the dataset before running LDA. For these topic modeling workflows, we took implementations of these algorithms that were available open source, and used them as components of the workflows. These included Mallet and OnlineLDA.

For community detection, the workflow uses the "bigclam" algorithm of the SNAP package and a visualization done with Gephi (both open source, the former part of the XDATA program).

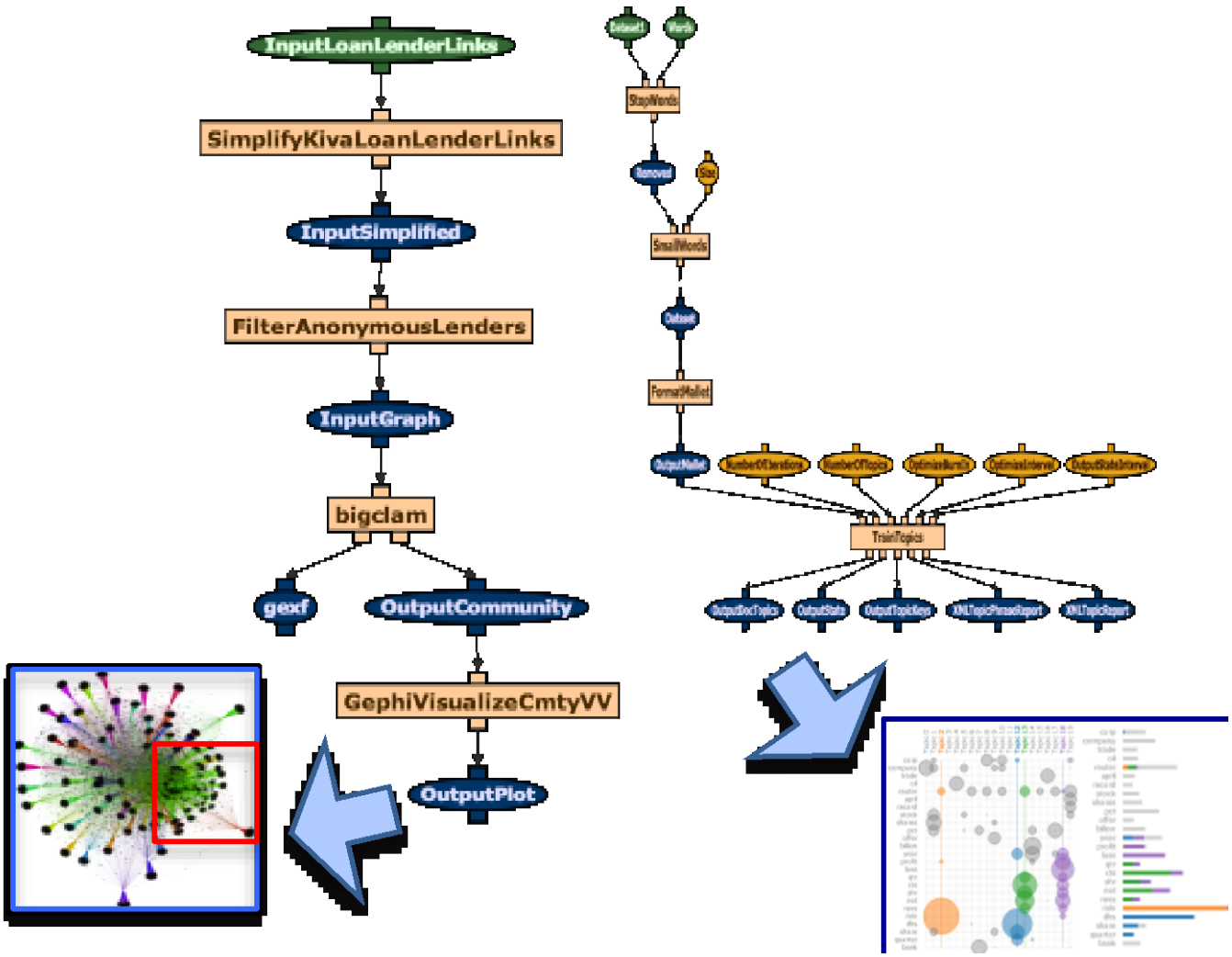


Figure 10(a): Workflows used to analyze the Kiva dataset with WINGS-OODT

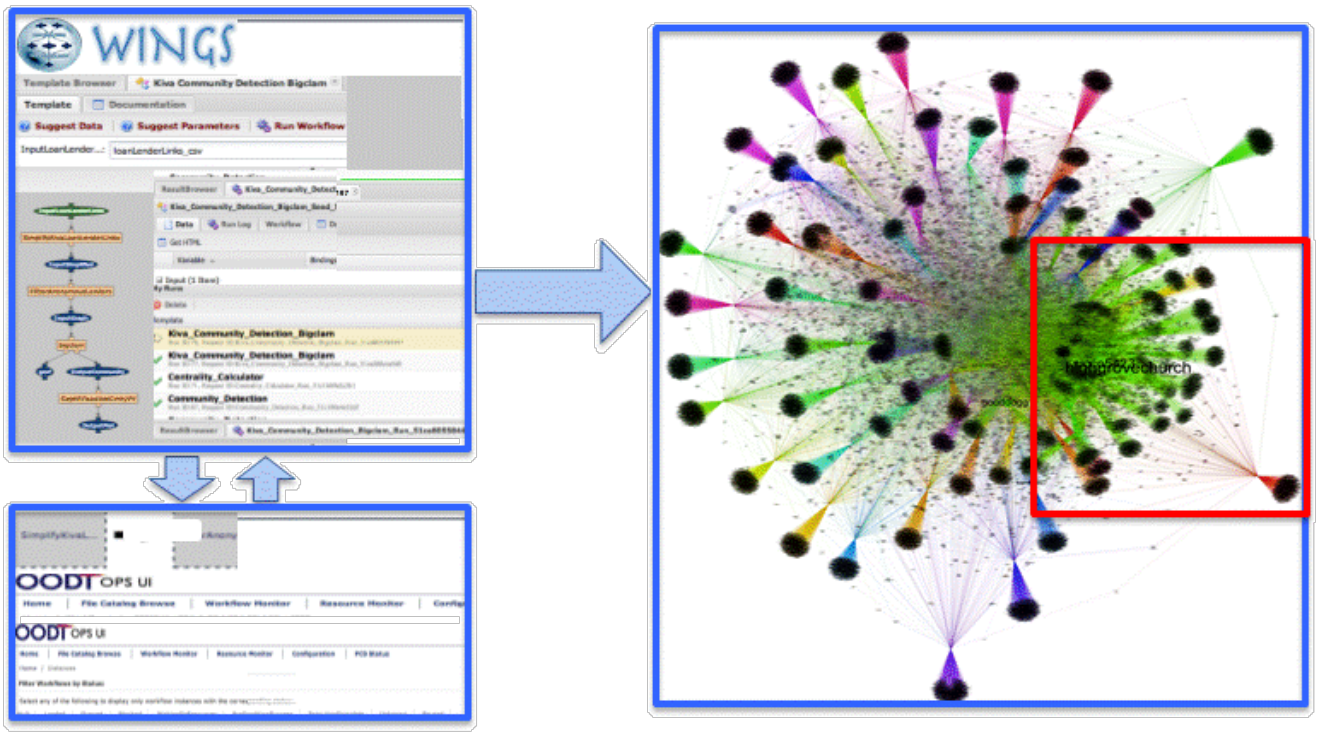


Figure 10(b): Kiva Dataset Analytics with WINGS-OODT Workflows

End users can easily use workflows like these to analyze their own data. Learning to use workflows does not require a lot of effort (e.g., a couple of hours for non-programmers that want to reuse workflows that are already in the system).

Figure 10(b) shows how end users interact with the WINGS interface to find a workflow for the kind of analysis they want to do with their data. Even users with no programming background find the structure of these workflows easy to understand. WINGS assists them by suggesting parameters as well as any ancillary data that the workflow may need (e.g., a dictionary). When the user is done specifying a workflow, WINGS elaborates it and submits it to OODT for execution. The user is then presented with the results and can browse any intermediate data as well as the provenance records for the results.

The visualization in the right-hand side of Figure 10(b) shows communities detected on Kiva Loan-Lender-Links data. The nodes shown are either lenders or loans, minus some nodes that we filtered out (eg anonymous). The visualization shows links between the first member of a community and the rest, resulting in the fan-like shapes. The size of the nodes is proportional to their centrality. The nodes are partitioned and colored according to modularity (an indication of the strength of the group). This visualization highlights, for example, that: 1) a group of nodes that forms a large community in green, 2) a number of nodes that are most salient in the data (eg, highgrovechurch, gooddog1, clive, richard,...).

The analysis results from faceted queries focused on regional distribution and country codes are shown in Figure 11.

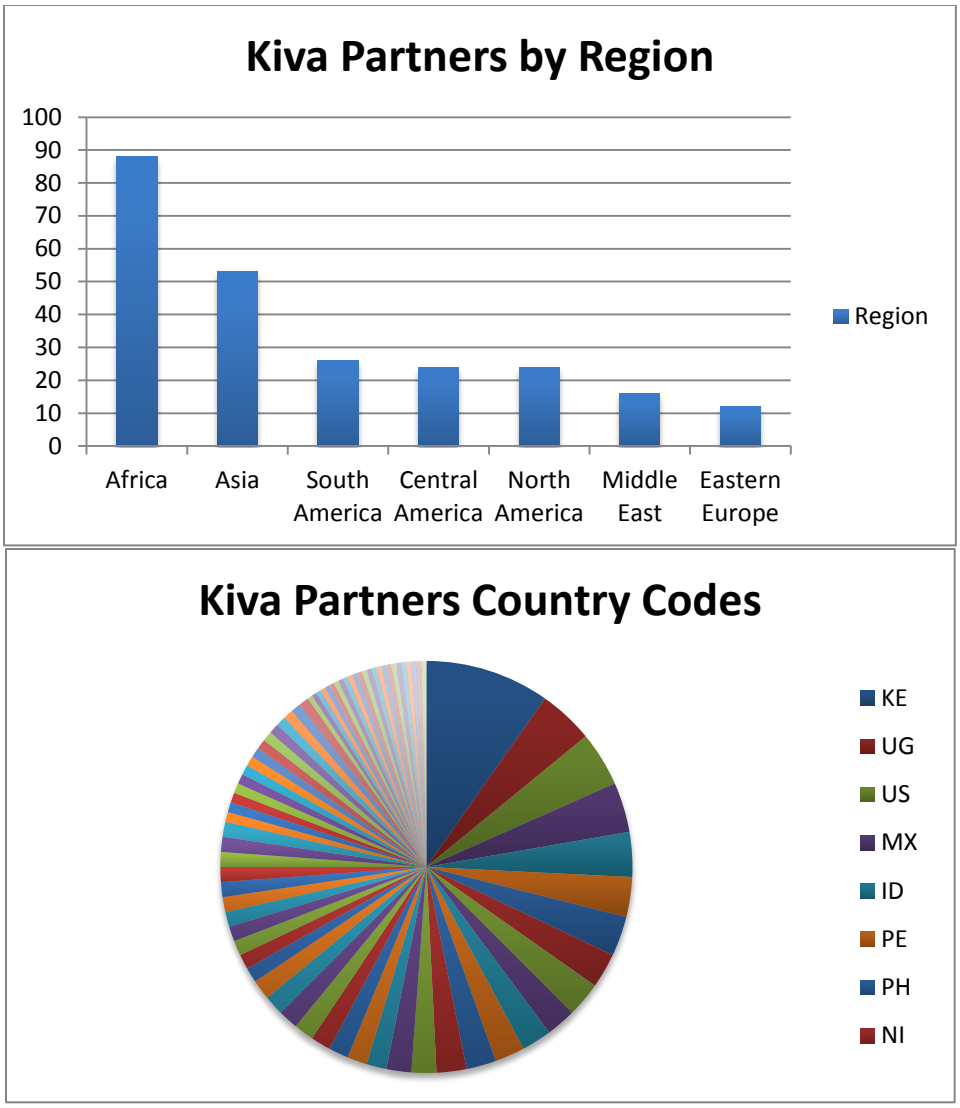


Figure 11. Kiva Partners by Region and Country Code

The time variation of loans from 2009 to late 2013 is shown in Figure 12. These plots were generated automatically using software we developed to plot from Apache™ Solr.

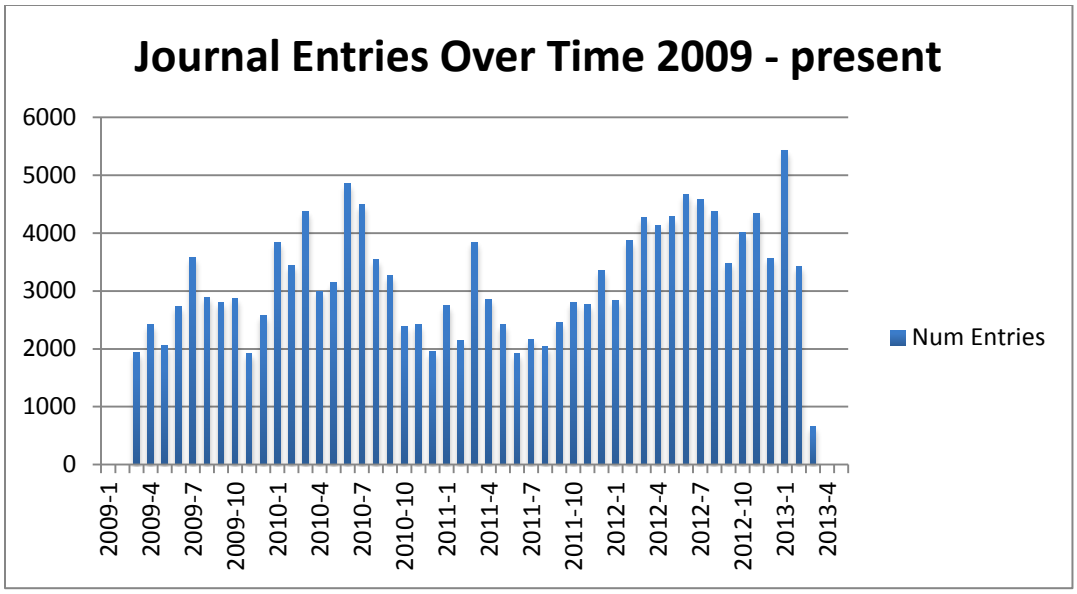


Figure 12. Kiva Variation of Journal Entries over Time

5.1.2 Analyses for Other Summer Challenge Problems

This involves working with Apache™ OODT and WINGS to perform the following analyses for the other 2013 Summer Challenges that we addressed. Two of the challenges were for Akamai internet traffic as represented in Traceroute and Edgescape data sets. In addition there was Twitter data and data from an instrumented exercise at the National Training Center (NTC) at Fort Irwin, California, known as the Global Intelligence, Surveillance, and Reconnaissance (GISR) dataset.

Akamai Traceroute and Edgescape data sets

The challenge problem described Akamai with this excerpt from Wikipedia (http://en.wikipedia.org/wiki/Akamai_Technologies):

"Akamai Technologies, Inc. is an Internet content delivery network headquartered in Cambridge, Massachusetts, in the United States. Akamai's network is one of the world's largest distributed-computing platforms, responsible for serving between 15 and 20 percent of all web traffic.

The Akamai Network is a distributed cloud computing platform that operates worldwide. It is a network of more than 250,000 servers equipped with proprietary software and deployed in more than 80 countries that relies on applied mathematics, computer networks and complex algorithms to help solve congestion, availability, performance and security problems on the Internet. These servers reside in more than 2000 of the world's networks monitoring the Internet in real time—gathering information about traffic, congestion, and trouble spots. Akamai uses this intelligence to optimize routes and replicate data dynamically to deliver content and applications."

There were two main categories of datasets associated with the Akamai challenge: the Classless Inter-Domain Routing (CIDR) dataset and the Traceroute/Edgescape data sets. We focused on the latter in order to exploit associated geographical information.

To that end, we did the following analytic steps.

- Followed Template ETL Process Defined with Kiva
  - Python extractors for Traceroutes and Edgescape data
  - Wrapped ETL in a workflow

    - No intermediate format used (i.e. JSON)
- Support Multiple Data Stores
    - Integration at the data store level instead
    - Minimize impact on existing tools
    - PostGIS
        - Widely supported open source GIS
        - Generating maps trivial
    - Solr Spatial
        - Newer capability blending existing capabilities and GIS
        - Relevancy of results can be ranked on spatial constraints

The primary results from this analysis were depictions of the geographic information associated with the edgescape data. Representative results are shown in Figure 13, which shows how different categories of internet service can be characterized based on distance from a certain location: here, by distance from Buffalo, NY.

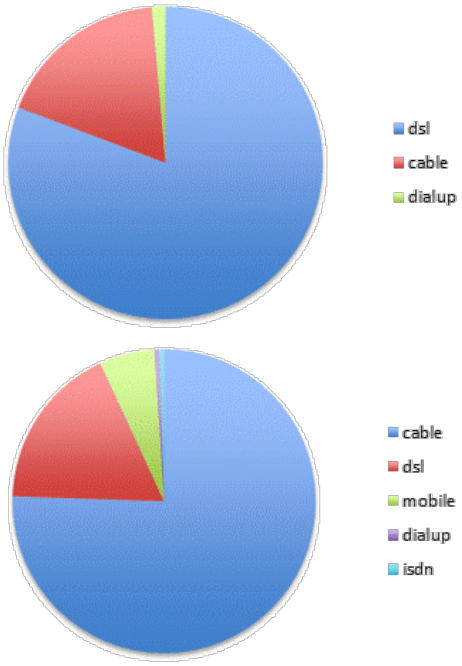


Figure 13. Akamai Edgescape network types within 25 km (top) and within 100 km (bottom) of Buffalo, NY.

In the process we learned the following.

- **There were numerous "Bad" records in the raw data**
- **It took many rewrites of the Python extraction code to account for the bad records and format details**

- **Work should be delegated to a higher level tool**
- **Analysis can drive what information should be extracted**
- **Many ways to represent the raw information**
- **Solr can handle GIS and faceted queries simultaneously**

These lessons will be valuable touch points going into subsequent similar analytic exercises.

Twitter data sets

The summer challenge description of Twitter follows.

"Twitter is a microblogging site and social networking tool that allows users to broadcast public text-based messages known as tweets. Tweets are limited to a length of 140 characters and often have much metadata associated with them. According to wikipedia, Twitter generates over 340 million tweets per day and has over 500 million registered users. Twitter provides access to their data at different pricing levels. In particular, they offer 1% of their tweets for free through a streaming API which was used a collection point from two vantage points."

This data often contains spurious information. There were two main datasets provided, one with more complete geographical information and additional derived fields.

We performed the following steps on these datasets.

- Ingestion Process
  - Python script to parse and POST documents
  - POST documents to Solr in batches (5-25K)
  - Apache™ OODT Workflow to manage 2000+ TSV files
  - Apache™ OODT OpsUI to monitor progress visually
- Downstream Integration
  - Twitter dataset 1&2 available via standard HTTP URLs
  - Wide variety of output formats (XML, JSON, CSV, Python/PHP/Ruby Array formats)
  - Subsets can be driven by queries and transformed at request time
  - New output formats generated based on XSLT or Velocity
- SolrCloud Configuration
  - Multiple Solr cores running in Jetty containers
    - SolrCloud managed by ZooKeeper
    - Sandbox machine: xdata.jpl.nasa.gov (16 core, 24 GB memory, 5TB disk)
  - Apache™ OODT Workflow:
    - 2 batch stubs @ capacity 8 concurrent threads
    - 2,300 jobs (1 per .tsv file)

We learned the following from performing the above processes. The top bullets refer to the size of the larger of the challenge datasets: 1.1 billion tweets. These were stored in the Solr database, here referred

to as “SolrCloud” because it was used with Zookeeper and special tweaking to achieve cluster coordination.

- SolrCloud for 1.1B tweets
  - Single Solr instance manifested issues at around 100 Million documents
  - SolrCloud = Solr + Zookeeper to create a coordinated cluster of Solr instances with the same schema
  - Index replicated/distributed across all nodes
  - Automatic periodic optimization of the index
- SolrCloud Performance 1.1B
  - ~3000 inserts / second across cluster
    - ~100 hours to ingest full 1.1B documents
  - ~150 GB index size/node for 250M documents
    - ~600 GB index size/node for full 1.1B documents
  - Most queries return in <1.5 seconds, repeat queries (cached) << 1 second
- SolrCloud Performance Geotwitter
  - ~3000 inserts / second across cluster
    - ~18 hours to ingest full 300M documents
  - ~2 GB index size/node for 2M documents
    - ~300 GB index size/node for full 300M documents
  - Most spatial queries return in <2 seconds, repeat queries (cached) << 1 second

GISR data sets

The problem description of the GISR challenge was: “The Global Intelligence, Surveillance, and Reconnaissance (GISR) dataset was captured with the purpose of being able to identify insurgent activities using a wide variety of sensor platforms. This dataset was captured during an exercise over the National Training Center (NTC) while insurgent activities were simulated amidst a background environment of normal activity. During the simulated insurgent activities, a variety of data collection sensors were deployed and collected massive amounts of data.”

MDA helped organize and was an active participant in the “GISR Working Group” established during the summer workshop. This was necessary because of sensitivity of some of the data.

We worked in a compressed timeframe of just a couple weeks in July 2013, as illustrated in Figure 14. The main MDA contribution was focused on finding correlations between data from LynxII (a ground-based fixed radio sensor) and VADER (an airborne radar sensor capable of accurate Ground Moving Target Indicator (GMTI) measurements).

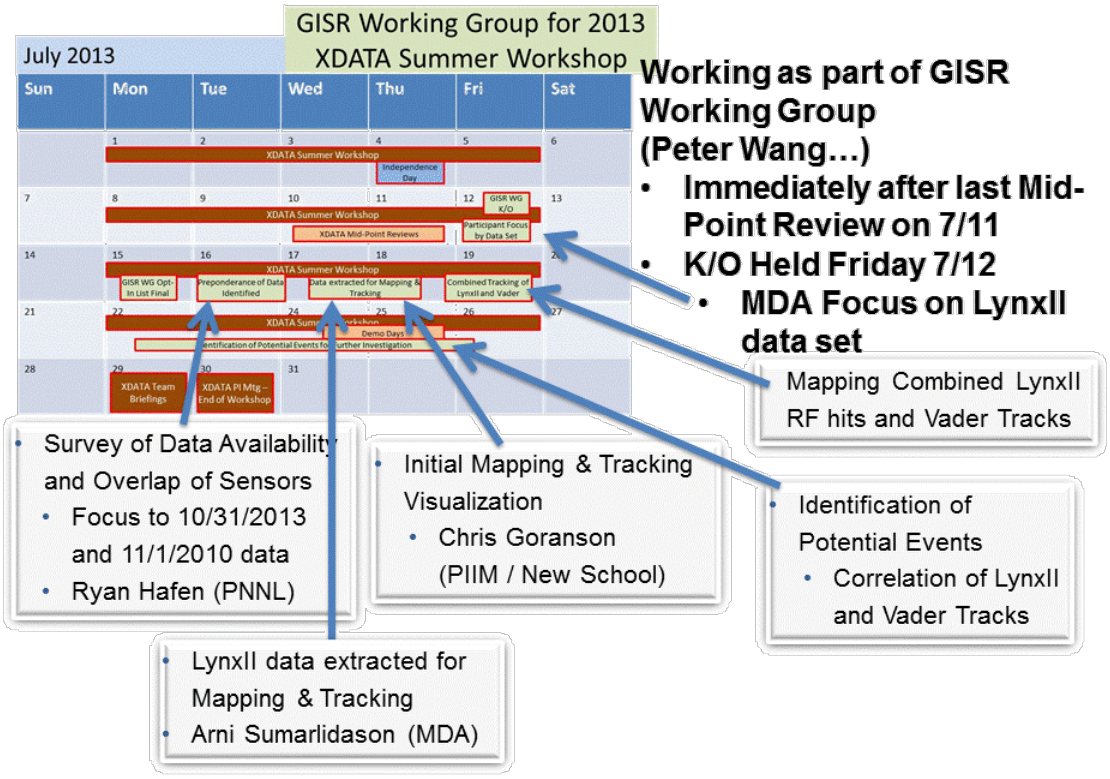
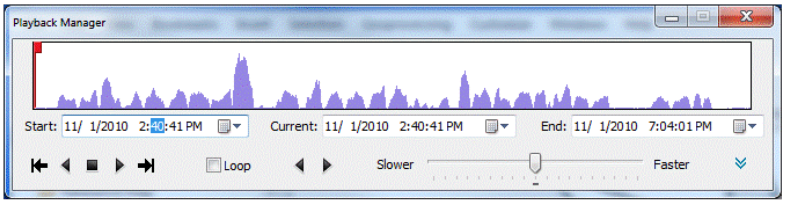


Figure 14. Participation in GISR Working Group Team Analysis

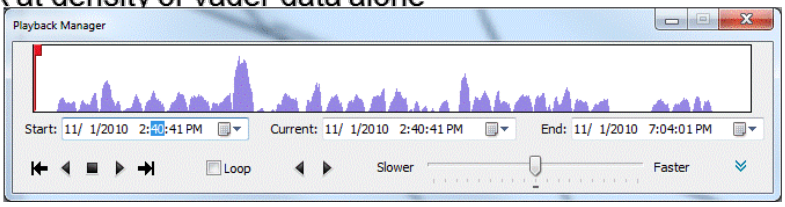
The steps in the correlation analysis are shown in Figure 15.

Steps in Analysis

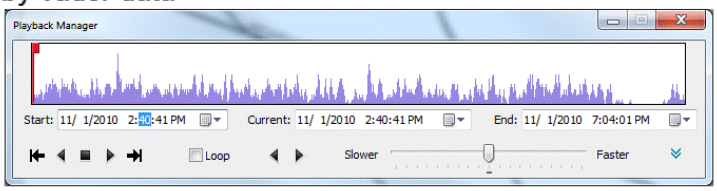
1. Look inside of the LynxII and Vader data during the identified dates: focus to 11/1/2010
2. Initial look at density of LynxII and Vader data
   - Tracks binned in 3 second increments
   - Density is dominated by Vader data



3. Look at density of Vader data alone



4. Look at LynxII data by itself – verifies that the total shape is dominated by Vader data



5. Focus initial inquiry on density maximum of combined LynxII and Vader data
   - For initial inquiry we want maximum chance of "seeing something"

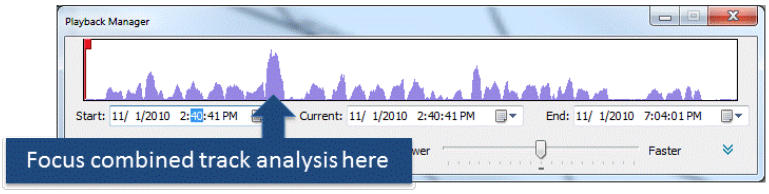


Figure 15. Steps performed in Analysis of GISR Data

A geographical look at the same data reveals an instance where a radio emitter moves from a roadway upon approach of a convoy, as illustrated in Figure 16. The LynxII RF emissions are indicated by the red circles. The Vader GMTI data is shown by the green dots.

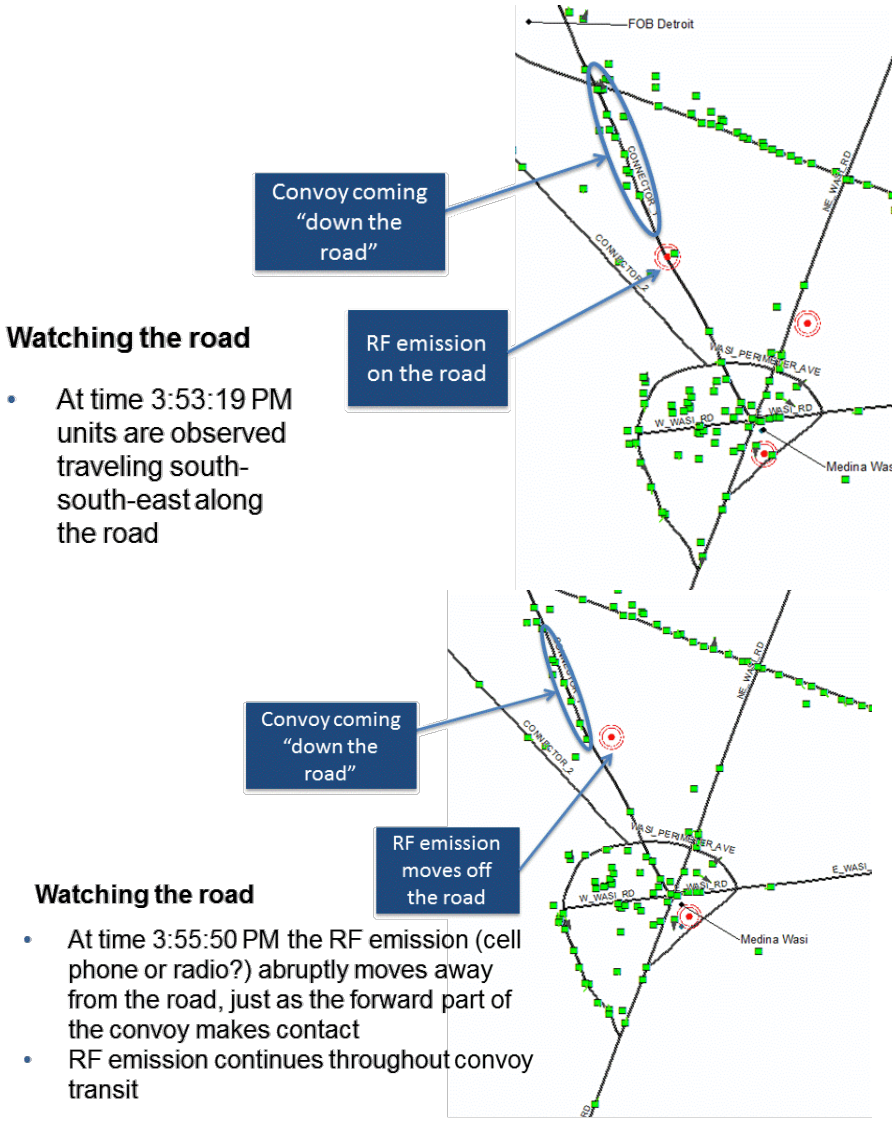


Figure 16. Geographic Depiction of Results of GISR Analysis. Two minutes elapse from the initial convoy approach (top) to when it passes the "observer" (bottom).

5.1.3 Extract, Transform, and Load (ETL) for the Summer Workshop Challenges

A survey of the many different but related approaches to performing the critical Extract-Transform-Load (ETL) processes by each performer team was done and lessons-learned presented at the PI meeting at the end of the summer workshop.

5.2 HIGHLIGHTS OF PARTICIPATION IN 2013 SUMMER WORKSHOP

The highlights for the MDA-JPL-ISI participation in the 2013 Summer Workshop are shown in Figure 17.

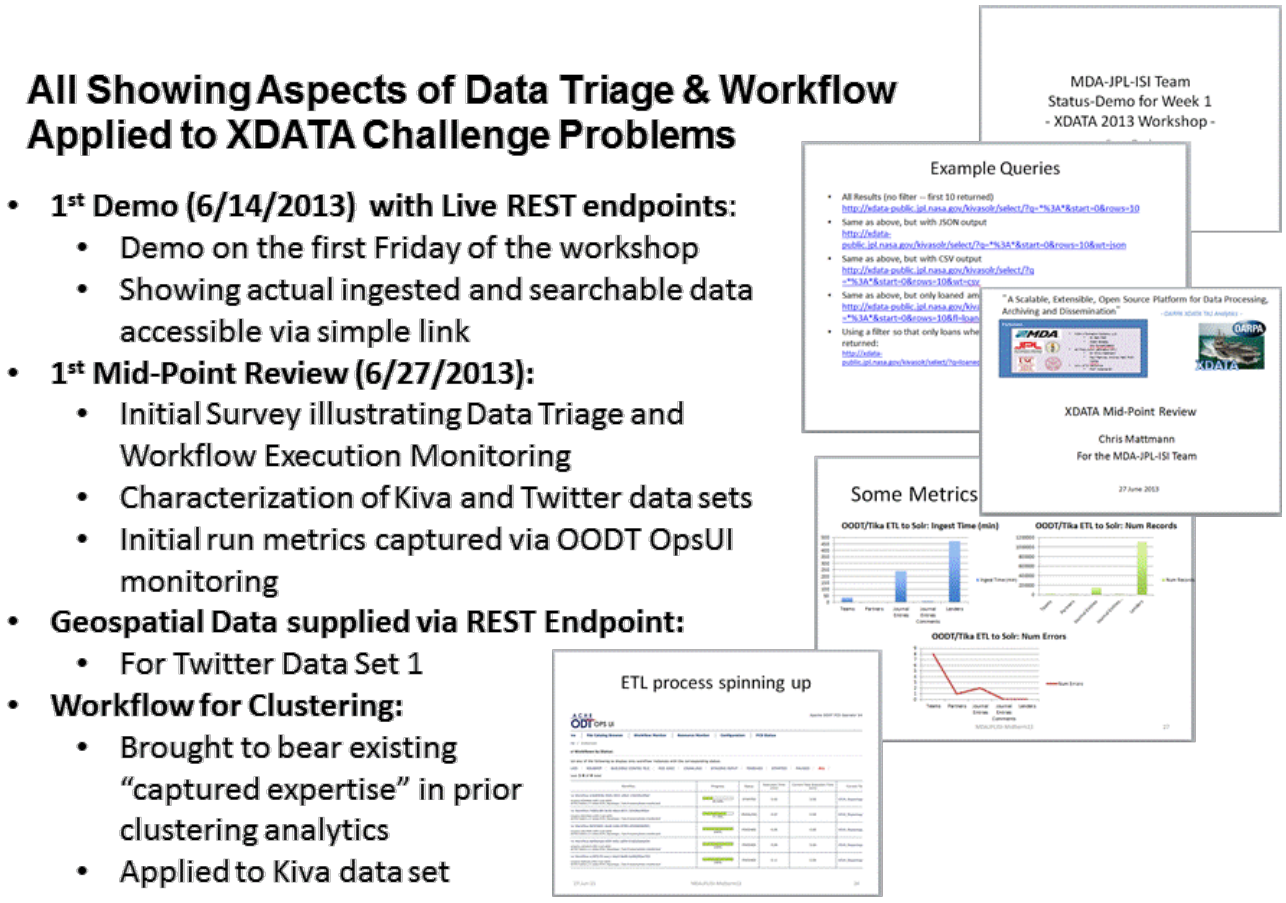


Figure 17 Highlights for MDA-JPL-ISI 2013 XDATA Summer Workshop

6.0 CONCLUSIONS

Beyond the illustrations of capability demonstrated early in the project, technical accomplishments during this project execution were i) analyses of the challenge problems during the 2013 Summer Workshop (Section 5.1) and ii) highlights of the MDA-JPL-ISI participation in the summer workshop (Section 5.2).

7.0 REFERENCES

[1] Blei, D., Ng, A., and M. Jordan. "Latent Dirichlet Allocation." Journal of Machine Learning Research, 3, pp 993–1022, January 2003.

[2] Carrington, L. C. et al. "How Well Can Simple Metrics Represent the Performance of HPC Applications?", Proceedings of the 2005 ACM/IEEE conference on Supercomputing, 2005.

[3] De Roure, D; Goble, C.;Stevens, R. "The design and realizations of the myExperiment Virtual Research Environment for social sharing of workflows". Future Generation Computer Systems, 25 (561-567), 2009.

[4] Furlani, T. R., Jones, M. D., Gallo, S. M., Bruno, A. E., Lu, C., Ghadersohi, A., Gentner, R. J., Patra, A., DeLeon, R. L., von Laszewski, G., Wang, F., and A. Zimmerman. Performance metrics and auditing framework using application kernels for high-performance computer systems. Concurrency and Computation: Practice and Experience, 25(7), 2013.

[5] Gil, Y., Groth, P., Ratnakar, V., and C. Fritz. "Expressive Reusable Workflow Templates." Proceedings of the IEEE e- Science Conference, Oxford, UK, pages 244–351. 2009.

[6] Gil, Y.; Deelman, E.; Ellisman, M. H.; Fahringer, T.; Fox, G.; Gannon, D.; Goble, C. A.; Livny, M.; Moreau, L.; and Myers, J. "Examining the Challenges of Scientific Workflows." IEEE Computer, 40(12), 2007.

[7] Gil, Y.; Ratnakar, V.; Kim, J.; Gonzalez-Calero, P. A.; Groth, P.; Moody, J.; and Deelman, E. "Wings: Intelligent Workflow-Based Design of Computational Experiments." IEEE Intelligent Systems, 26(1). 2011.

[8] Gil, Y.; Gonzalez-Calero, P. A.; Kim, J.; Moody, J.; and Ratnakar, V. "A Semantic Framework for Automatic Generation of Computational Workflows Using Distributed Data and Component Catalogs." Journal of Experimental and Theoretical Artificial Intelligence, 23(4), 2011.

[9] Hauder, M., Gil, Y. and Liu, Y. "A Framework for Efficient Text Analytics through Automatic Configuration and Customization of Scientific Workflows". Proceedings of the Seventh IEEE International Conference on e-Science, Stockholm, Sweden, December 5-8, 2011.

[10] Hauder, M.; Gil, Y.; Sethi, R.; Liu, Y.; and Jo, H. "Making Data Analysis Expertise Broadly Accessible through Workflows." Proceedings of the Sixth Workshop on Workflows in Support of Large-Scale Science (WORKS'11), held in conjunction with SC 2011, Seattle, WA, 2011.

[11] Hoffman, M., Blei, D., and F. Bach. "Online Learning for Latent Dirichlet Allocation." NIPS, 2010.

[12] Hutter, F., Xu, L., Hoos, H. H., and K. Leyton-Brown. "Algorithm Runtime Prediction: The State of the Art". Available from arXiv:1211.0906.

[13] Langford, J. Vowpal Wabbit. https://github.com/JohnLangford/vowpal_wabbit/ 2011.

[14] Mattmann, C. A., et al. "A reusable process control system framework for the orbiting carbon observatory and NPP Sounder PEATE missions." Third IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT), 2009.

[15] Mattmann, C. A., and J. Zitting. "Tika in Action." Manning Publications, 2011.

[16] McCallum, A. K. "MALLET: A Machine Learning for Language Toolkit." http://mallet.cs.umass.edu. 2002.

APPENDIX A – THE ZERO DARK THIRTY USE CASE FOR APPLYING OODT TO "UNLOCKING BIG DATA"

This was published as **Unlocking Big Data**. Park, Samuel L.; Mattmann, Chris A. In July-August issue of *Geospatial Intelligence Forum* (Vol 11, #5, pages 25-26) http://issuu.com/kmi_media_group/docs/gif_11-5_final/27.

Going beyond the "Zero Dark Thirty" Scenario, Data Triage can provide the key to understanding massive amounts of newfound information.


Samuel L. Park


Chris A. Mattmann

The Navy SEALs proceeded with their mission, with obvious urgency, floor by floor, sack by sack, gathering potentially valuable material along the way, affixing information stickers and stacking it all for team extract.

As portrayed in the movie *Zero Dark Thirty*, SEAL Team Six had just flawlessly executed the raid on Osama bin Laden's Abbottabad residence. But the mission was not over—not in a site worthy of Sensitive Site Exploitation attention. They also had to emerge with the trove of potential information that could be there. They egressed beyond the building perimeter, not resting until unburdening themselves at a rendezvous and debriefing area.

While maintaining chain of custody, the unloaded, labeled material was transitioned to responsible analysts, who then faced their own daunting challenges. What kind of data was there? What was it? How much of it was there? Was it connected to other data on the web or elsewhere? Most of all, what did it mean?

This *Zero Dark Thirty* scenario motivates consideration of the technical and operational challenges that face time-constrained intelligence extraction from newfound data troves of unknown size, characteristics or content. Analysts must deal with voluminous multi-terabyte sized hard drives and gigabyte sized cellphones, each likely containing links to a much larger world of legacy databases, social media and the world wide web. A city of Sherlocks would be needed just to eyeball the data, much less to draw clever inferences from it. Welcome to today's world of big data.

Some quick assessment of the whole situation is needed, and for big data this ultimately comes down to having a smart way to do data characterization at ever finer levels. This data triage, important in itself, further provides a quick first characterization and initial disposition of the newfound data. It thereby facilitates other exploitation of opportunistically discovered big data.

New tools such as Apache Object Oriented Data Technology (OODT) and Apache Tika make this data triage more rapid, effective and easy to implement, ultimately changing a practically infeasible challenge into something matter-of-course. OODT and related capabilities can be quickly brought to bear, in combination with legacy and developing analytics. The resulting system can be ramped up quickly to operational scope and scale in order to address an urgent problem, all the while maintaining key linkages with subject matter experts.

New programs sponsored by the Defense Advanced Research Projects Agency (DARPA), especially the XDATA initiative, support these efforts and provide new capabilities for government, non-profits and commercial enterprises.

Big-Data Triage

Big data is big in at least three ways—in volume, velocity and variety, sometimes called the "3 Vs" of big data. The initial challenge with a new cache of unknown data is its potential variety. It could be virtually anything that bits can be made to represent. Beyond the data's size, its structure/architecture, volume and connectedness provide the first meaningful characterizations of it. The sheer number and continuing proliferation of file formats is by itself a significant challenge, even without considering encryption.

According to FileExt, the file extension information website (http://fileext.com), there are more than 50,000 file types currently in use, and growing. Many of them require specialized, often proprietary, software to process and understand. The challenge becomes vastly more complicated as links to other data are discovered, which themselves can be linked to yet more. Sophisticated graphical mod- els are needed just to capture the general overall structure. Further, in many sensor and social media applications, the large volume of data arrives in continual repetition. This velocity requires creating special storage and/or filtering/synthesizing of the data.

At the start of this ever-expanding domain is the challenge of "data triage": How does one go swiftly from the first characterizations to subsequent ones? NASA's Jet Propulsion Laboratory (JPL) has been a pioneer in developing computational approaches that can handle such situations.

OODT, a set of related collection and archiving tools, is one such innovation. OODT has been proven over the last decade for its use on a number of earth observing, climate and science missions. It was transitioned by NASA to the Apache Software Foundation (ASF) in 2010, the first NASA software project to do so. Apache OODT became a top-level ASF project in January 2011, and continues to rapidly develop new capability, specifically for applications in big sci- ence and in medicine.

OODT provides tools that quickly wrap legacy or new applications, enabling them to be connected to each other, to data sources and to end-users with a minimum of fuss and expenditure. Apache OODT was recently selected as part of a proposal by MDA Information Systems (MDA), teamed with JPL, as a big data technology in DARPA's new XDATA initiative.

Some tools associated with OODT are ideal for "automated meta- data extraction," which essentially provides the first characterization of newfound data, including automated file-type detection over an extremely broad range of file types. The Apache Tika framework delivers this capability seamlessly. This was exercised as an example application during the XDATA program kickoff in January, where a set of example big data was "crawled," characterized and interfaced with an iPhone.

The results of this exercise are applicable to a number of domains, including big science, medicine and geospatial intelligence. For example, usage of geolocation data in social media min- ing and analytics presents big-data challenges, which can now be mitigated using OODT- and Tika-based data acquisition, metadata extraction, crawling and MIME type detection.

Although OODT and Tika are admirably adaptable, they exist in a universe of software having many capable big-data alternative analytics, many of them having higher performance in niche processing roles. The special role of OODT is in taking the first whack at things, figuring out what data is there, wrapping legacy analytics, connecting all parties together and expanding to scale. After this is done, more specialized applications may be brought to bear. This is essentially the role called for in data triage.

The wrapping of analytics is not only quick, but also effective by several measures. By using the legacy code itself, in an unaltered form that has been intimately vetted by subject matter experts (SMEs), it reduces errors that often arise from recoding. These errors are often extremely difficult to ferret out and can result in costly erroneous results and the follow-on consequences of them being used. OODT's modularity also allows each SME's analytics to be included in a parallel development process, further reducing calendar time to getting a result.

Data triage comes in at least two parts: first, the quick characterization of data, then, the disposition of that data to other analytics. The second step calls for invoking workflows to accomplish these tasks. For big data, because of its size, such subsequent workflows often take some time to execute, though preliminary results are a common and quite useful characteristic as well. The coordination of these analytic steps is greatly facilitated by new visualization approaches.

DARPA's XDATA program has anticipated these needs. Of the 24 XDATA performers, eight are doing visualization development. MDA and JPL are working with these visualization performers on providing new, more approachable ways of taking advantage of these big-data analytics—ways that do not require command line expertise to function.

FREE AND OPEN SOFTWARE

Lastly, the OODT and Tika software themselves are free and open, meaning there is open knowledge of what is in the software and how it functions, there is an extended community of contributors and users, and the body of software itself is a ready source of "capital" for forward-looking business, non-profit organizations and government. This openly available new capital is a resource that all stakeholders can use to advantage or, conversely, ignore to their disadvantage.

Answering the data triage challenge is the start to answering a whole lot more about big data. Leveraging its proven pedigree in big science, new tools associated with Apache OODT can be brought to bear in the geospatial community. OODT and related analytics can be brought to bear adaptively, scaled to operational scope, all the while using a wrapping approach that maintains maximum linkage with experts in the field.

What is your geospatial big-data challenge? Perhaps you need to triage an abundantly endless collection of geo-tagged social media feeds? Or quickly assess information from your UAV and satellite imagery feeds? When faced with your next big-data opportunity, utilize these new open source tools that can make practical use of this data in a hurry: Link it in, scale it up and use it to the max.

APPENDIX B – TIME-BOUND ANALYTIC TASKS ON LARGE DATASETS THROUGH DYNAMIC CONFIGURATION OF WORKFLOWS

Published as: **Time-Bound Analytic Tasks on Large Datasets through Dynamic Configuration of Workflows.** Gil, Y.; Ratnakar, V.; Verma, R.; Hart, A.; Ramirez, P.; Mattmann, C.; Sumarlidason, A.; and Park, S. L. In *Proceedings of the Eighth Workshop on Workflows in Support of Large-Scale Science (WORKS), held in conjunction with ACM Supercomputing 2013*, Denver, Colorado, 17 November 2013.

Time-Bound Analytic Tasks on Large Datasets through Dynamic Configuration of Workflows

Yolanda Gil
Varun Ratnakar
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
gil@isi.edu, varunr@isi.edu

Rishi Verma, Andrew Hart,
Paul Ramirez, Chris Mattmann
NASA Jet Propulsion Laboratory
4800 Oak Grove Drive,
Pasadena, CA 91109
{rishi.verma, andrew.f.hart,
paul.m.ramirez,
chris.a.mattmann}@jplnasa.gov

Arni Sumarlidason
Samuel L. Park
MDA Information Systems LLC
820 West Diamond Ave., Suite 300
Gaithersburg, MD 20878
arni.sumarlidason@mdaus.com
sam.park@mdaus.com

ABSTRACT

Domain experts are often untrained in big data technologies and this limits their ability to exploit the data they have available. Workflow systems hide the complexities of high-end computing and software engineering by offering pre-packaged analytic steps and combining them into multi-step methods commonly used by experts. A current limitation of workflow systems is that they do not take into account user deadlines: they run workflows selected by the user, but take their time to do so. This is impractical when large datasets are at stake, since users often prefer to see an answer faster even if it has lower precision or quality. In this paper, we present an extension to workflow systems that enables them to take into account user deadlines by automatically generating alternative workflow candidates and ranking them according to performance estimates. The system makes these estimates based on workflow performance models, and uses semantic technologies to reason about workflow options and their quality. Possible workflow candidates are presented to the user in a compact manner, and are ranked according to their runtime estimates. We have implemented this approach in the WOOT system, which combines and extends capabilities from the Wings semantic workflow system and the Apache OODT Object Oriented Data Technology and workflow execution system.

Categories and Subject Descriptors

C. Computer systems organization, D.2 Software engineering, D.2.10 Design.

General Terms

Design, Performance, Human Factors.

Keywords

Workflows, semantic workflows, performance, Wings, OODT.

1. INTRODUCTION

Big data is pushing the boundaries of data volume, velocity, and complexity, and generating much interest in developing software systems that can not only meet the scale of increasing information from higher resolution scientific instruments, sensors, business and financial systems, and the like; and also that meet the ability to deal with complex representations of the information, including meta-data or ("data about data") – often including hundreds to thousands of attributes and rich descriptions of information that help to locate it; understand its provenance, and reason using it both retrospectively and prospectively.

Workflow systems describe the common tasks that big data producers, and algorithm developers execute that transform information throughout its lifecycle from data production; to processing/transformation and ultimately to its distribution [Woollard et al 2008]. Workflows have the advantage of being summarized descriptions of complex tasks and thus they can be more easily explained scientists, intelligence analysts, business analysts, and other users.

It is possible to design alternative workflows for the same task that have very different performance, particularly for big data. For example, many different workflows can be created using different algorithms to detect popular topics in a large collection of documents (e.g., news articles, tweets, etc). An end user may need a workflow to perform that task within a certain deadline, or have a desired accuracy. When datasets are very large, and the performance varies widely depending on many metadata characteristics and parameter settings, how can an end user compare and select among possible alternative workflows?

In this paper we describe an approach to enable end users to get workflow solutions that meet their performance requirements, in particular runtime deadlines. We leverage: 1) semantic workflows to automatically generating candidate workflows from a given specification of their inputs, and outputs, and allowing a user to evaluate those workflows in various scenarios, 2) workflow execution with integrated data and metadata management and provenance recording, and 3) learning predictive performance models from prior workflow executions.

Our approach allows a *workflow designer* to create abstract workflow templates that can be run using different application algorithms, and provide training data (e.g., sets of inputs and outputs) which are used in a learning phase to create a performance model for each workflow under different datasets and parameter settings. When a *workflow user* provides a set of performance requirements, the system automatically generates possible candidate specializations of the abstract template and uses the learned performance model to rank those candidate workflows.

We have implemented this approach in WOOT, a system that combines the semantic workflow capabilities of WINGS [Gil et al 2011a; Gil et al 2011b] and the metadata extraction and provenance tracking capabilities of the Apache OODT Object Oriented Data Technology and workflow execution system [Mattmann et al 2009].

The rest of the paper is organized as follows. The next section motivates through examples the needs of end users as they confront the analysis of big data at scale in the face of many alternative algorithms, implementations, and methods. Section 3 surveys related work. Section 4 introduces our approach, defining five key capabilities needed. Section 5 explains the architecture of WOOT that combines and extends Wings and OODT to achieve those capabilities, walking through examples along the way. Section 6 discusses additional aspects of this problem that would require extensions to our work to date. Finally, we present conclusions and future work.

2. MOTIVATION

Big data requires a range of expertise that very few people have. End users have a deep understanding of their domain and the questions they want answered from the vast amounts of data, but do not have the range of skills required to analyze it. Our goal is to empower end users with the ability to analyze their data, and workflows can help in many ways.

Consider a social scientist who has access to large amounts of social media data, such as tweets. He is interested in understanding the dynamics of followers based on affinity to topics of tweets. Another example would be a communications student doing a thesis on what groups of teenagers discuss particular types of topics in social media. Consider also an epidemiologist trying to understand the spread of tuberculosis from social media data. Finally, consider a historian who, as suggested by http://programminghistorian.org, has a large set of documents such as multi-year newspaper records (http://dsl.richmond.edu/dispatch/) or decades worth of daily diaries (http://historying.org/2010/04/01/topic-modeling-martha-ballards-diary/). She would like to know the topics that were being discussed at any given time and how they changed over the years particularly in response to known historical events.

These are all examples of end users, who may be aware that there are topic modeling techniques from natural language processing that they could apply but do not know where to start.

Workflows provide a mechanism to capture state-of-the-art multi-step methods that experts would define for a particular task, and make those methods available to non-experts. Figure 1(a) shows an example workflow for topic modeling taken from [Hauder 2011a]. The workflow starts removing stop words (e.g., punctuation) and short words (e.g., "the", "of", "and"), and then converts the data to a format that can be used by a state-of-the-art topic modeling algorithm that uses Latent Dirichlet Analysis (LDA) [Blei et al 2003]. This particular workflow was used with little training by high-school students to analyze twitter data [Hauder et al 2011b]. Experts can define workflows and share them with others through repositories [De Roure and Goble 2009] or as open web objects [Garijo and Gil 2011].

However, many different algorithms and approaches exist for topic modeling, and many alternative implementations of those algorithms exist, each requiring different preparation steps to format the data, offering different parameters, and claiming efficient performance. Therefore, many different workflows might be available. Figure 1 shows alternative workflows that use different algorithms and implementations for LDA. An end user would wonder which implementation would give them an answer faster, and what workflows will give the best answer (a "good" quality answer) within the time bounds.

Figure 1(a) shows a workflow (WF-LDA-MALLET) that uses a popular implementation for LDA in the MALLET package [McCallum 2002]. The site http://mallet.cs.umass.edu/topics.php indicates:

> *"The MALLET topic model package includes an extremely fast and highly scalable implementation of*

Gibbs sampling and efficient methods for document-topic hyperparameter optimization."

An analogous workflow (WF-LDA-TMT) could be built with the TMT software at http://www-nlp.stanford.edu/software/tmt/tmt-0.4/, which also implements LDA.

Figure 1(b) shows a workflow (WF-OLDA) that uses online LDA [Hoffman et al 2010], an algorithm for online learning that builds the topic models as it processes documents incrementally:

> *"Online LDA is based on online stochastic optimization with a natural gradient step […]. It can handily analyze massive document collections, including those arriving in a stream."*

There are several implementations of this algorithm leading to different workflows: gensim [Řehůřek 2009] (WF-OLDA-GENSIM) and Vowpal Wabbit [Langford 2011] (WF-OLDA-VW), both in Python.

Figure 1(c) shows another workflow that uses a parallel version of the LDA algorithm [Wang et al 2009]:

> *"PLDA can be applied to large, real-world applications and achieves good scalability."*

There are actually two versions of this workflow: one with an MPI implementation (WF-PLDA-MPI) and a MapReduce implementation (WF-PLDA-MR).

Finally, Figure 1(d) shows a workflow (WF-LDA-VIZ) that first creates a topic model and then generates a visualization of it. In addition, it includes a sampling step at the beginning, which can be used to reduce execution time by reducing the size of the input dataset.

To complicate matters, not only are the above LDA algorithms very different but they use a different set and number of input parameters and have a very different performance depending on the value of the parameters. For example, although Mallet LDA and online LDA both have a parameter to indicate the number of iterations, Mallet LDA has another 4 parameters (e.g., optimization bounds and optimization interval) that are different from online LDA's other 2 parameters (e.g., batch size).

All the algorithms have a parameter to specify the target number of topics desired. They also share another parameter that specifies the number of iterations of the algorithm. This parameter in particular greatly affects the performance of the algorithm in terms of execution time. Parallel LDA has a parameter to specify the number of processors to use. Below a certain number, performance is likely to suffer from the communication overhead.

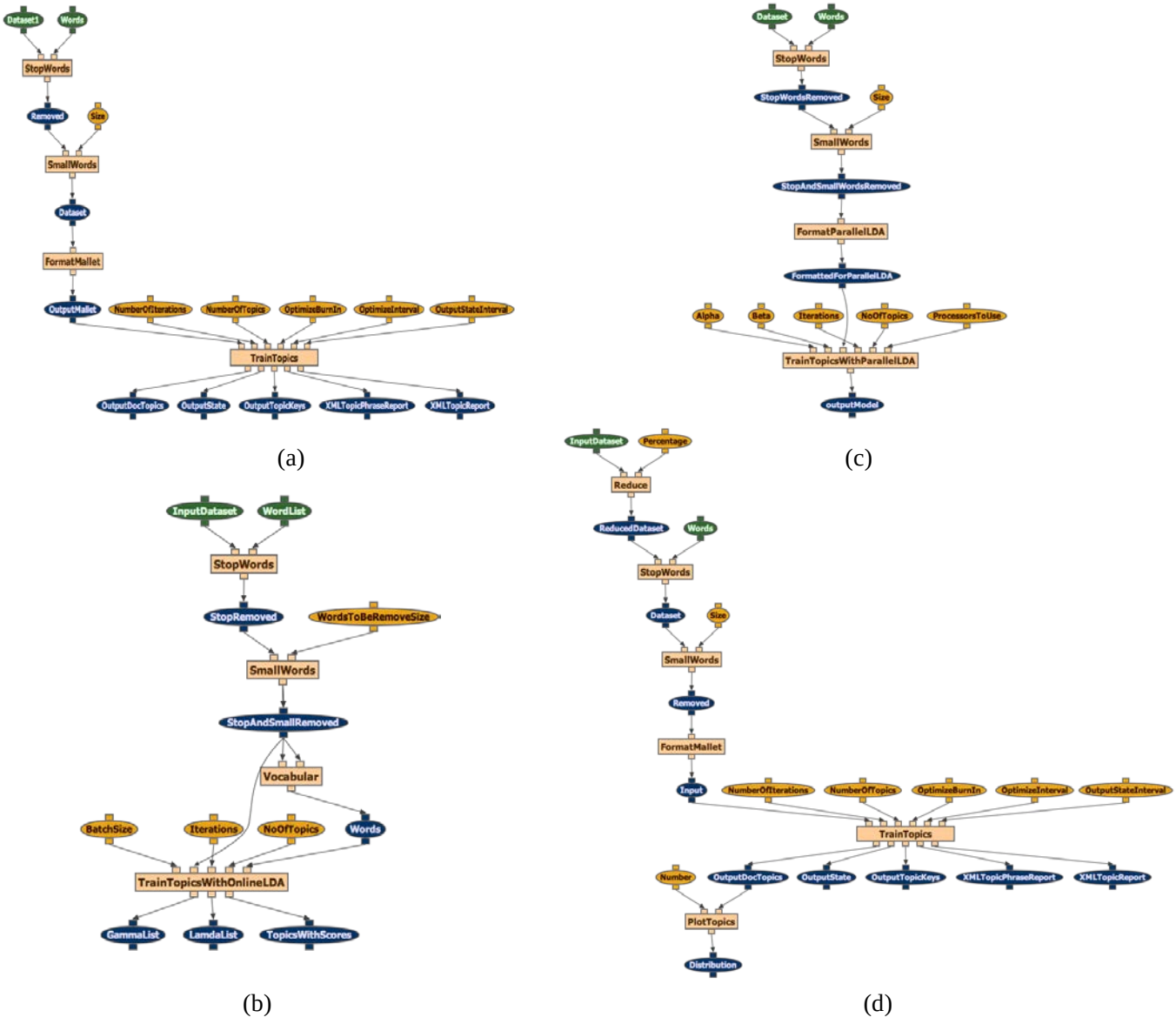


Figure 1: Alternative workflows for topic modeling using LDA: a) Mallet implementation, b) Online LDA, c) ParallelLDA, and d) an LDA implementation that includes a final visualization step and an initial sampling of the input dataset.

So suppose a user has a very large dataset, and wants to obtain topics within a certain time bound (e.g., 1 hour). What would be the best workflow for them? This is the problem we aim to address in this paper.

3. RELATED WORK

Retrieving workflows from repositories has been a topic of active research. Some approaches look at matching based on tags [Goderis et al 2007], the shape of the workflow graph [Goderis et al 2008]. In our own work, we investigated the retrieval of workflows based on high-level requests, such as finding workflows that generate a desired type of result or workflows that are appropriate to process a given type of data. Furthermore in the Apache OODT workflow system [Mattmann et al 2009] workflows are retrieved based on a series of dynamic multi-valued metadata – the same metadata that is used in file cataloging; metadata extraction and curation and in resource management. This metadata can correspond to workflow status (e.g., FINISHED, EXECUTING; or FAILED); current task wall clock time or workflow wall clock time; workflow id; workflow instance id; and other information. We leverage this workflow metadata in our approach for workflow selection detailed in section 5. In our own work, and more broadly within the community retrieval of workflows based on performance bounds has not been explored before.

There is a vast literature on performance modeling that is directly relevant to this question. Performance

modeling relies on collecting performance data for a given algorithm or code with different input datasets and parameter settings, then building a predictive model that can be used to estimate runtime for new datasets (see [Hutter et al 2012] for an overview). Algorithms are often considered to be "workflow blocks" and their predicted performance based on different input sizes can be used to improve the allocation of resources [Miu and Missier 2012]. In our work, we consider the performance of the entire workflow rather than individual codes. Because our focus is on extending workflow systems to generate and use workflow performance models, our models are quite simple and can be extended in future work to incorporate techniques from this body of work.

Workflow performance is often used to compare across execution infrastructures [Montagnat et al 2010; Vahi et al 2012]. Workflow runtime is one of many metrics that must be taken into account, others can include resource utilization and reliability [Furlani et al 2013; Carrington et al 2005]. Predicting workflow performance is key for resource reservation and resource allocation. All these sophisticated metrics and performance measurements could be used to extend our approach and provide end users with additional tradeoffs.

In prior work, we investigated performance/quality tradeoffs in the context of biomedical image analysis [Kumar et al 2009]. The work focused on the creation of the performance models, which expert high-end computing users would then inspect and based on them decide how to set up parameters for a new dataset. However, there was no interaction with an end user. In addition, the focus was on the selection of parameters, while we focus here on the selection of both algorithms and their implementations in addition to parameters.

4. APPROACH

Our approach is to:

1. *Learn workflow performance models*: For every workflow, the system learns performance models using training datasets of different sizes and characteristics as well as using different parameter settings.
2. *Allow users to specify their requirements and constraints*: Users should be able to specify the desired task and request an answer within a specific time bound.
3. *Automatically generate candidate workflows*: Given the user task and time bound, the system automatically retrieves all the relevant workflows and instantiates them with possible parameter values.
4. *Rank candidate workflows*: The system uses the workflow performance models to rank the candidate workflows.
5. *Present users with options*: The system selects one or more workflows to run, and presents the user with other possible workflows.

To achieve these capabilities we combine two complementary workflow systems, Wings [Gil et al 2011a; Gil et al 2011b] and Apache OODT [Mattmann et al 2009], and augment them with new features.

Wings is a semantic workflow system that supports the specification of abstract workflow templates that include classes of steps (e.g., LDA) that can be specialized to specific algorithms or implementations (e.g., Parallel LDA using MPI). Wings uses workflow reasoning algorithms that take an abstract workflow and automatically generate workflows of executable application codes that can be submitted to a workflow execution engine.

OODT is a distributed data management and processing framework, with components to extract metadata and do profiling, and with distributed execution of workflow components that can be managed from the workflow execution engine.

OODT greatly facilitates learning performance models (item 1 above), since it can extract metadata characteristics upon workflow execution as previously discussed in Section 3 – these characteristics are represented using OODT's canonical key, multi-valued metadata representation.

Wings facilitates the generation of candidate workflows for user requirements (items 2 and 3 above), since it has a workflow matching engine that can retrieve relevant workflows given a high-level specification of a user's task [Bergmann and Gil 2012] as well as an algorithm to search the space of possible workflow instantiations [Gil et al 2011a; Gil et al 2011b].

In addition to integrating Wings with OODT to take advantage of these capabilities, we developed new functionality to rank workflows (item 4 above) and to present the user with options (item 5).

In the next section, we describe our implementation of this approach.

5. WOOT: The WINGS/OODT Workflow Recommender

Figure 2 shows a high-level overview of the architecture of the system, highlighting in dashed lines the new capabilities in WOOT. In the left-most portion of the diagram, a Big Data expert interested in making available the necessary information for workflow selection and identification provides a particular set of candidate workflows, along with a set of training data culled from executing the workflow on large numbers and variations of inputs. These inputs may also include relative quality assessments, allowing evaluation of the results of the provided candidate workflows on training data. These form the basis for ranking and allowing selection of the appropriate workflow for the ultimate system end-user shown in the right-most portion of the Figure 2.

In the middle most portion of the diagram, the library of candidate workflows is made available to both Wings (in the upper most portion of Figure 2), and to OODT (in the bottom most portion of Figure 2). These workflows are used as input to Wings in order to develop large numbers of candidate inputs, and variances in the numbers of inputs to the workflows. After this initial pre- execution step is performed, the workflows are executed in the OODT

workflow execution engine, which in turn stores provenance metadata about the workflow such as its start/stop wall clock time at a per task level and per workflow level, as well as specific workflow instance metadata (e.g., the input parameters that were provided). This information is stored in the data and provenance catalog for OODT shown in the bottom middle portion of Figure 2.

In the upper right portion of Figure 2, the OODT provenance and data catalog are mined to assess workflow performance and ultimately rank the candidate workflows, and to provide this information to the end-user in the upper right portion of Figure 2.

The next sections describe our implementation of each of the five aspects of our approach listed above.

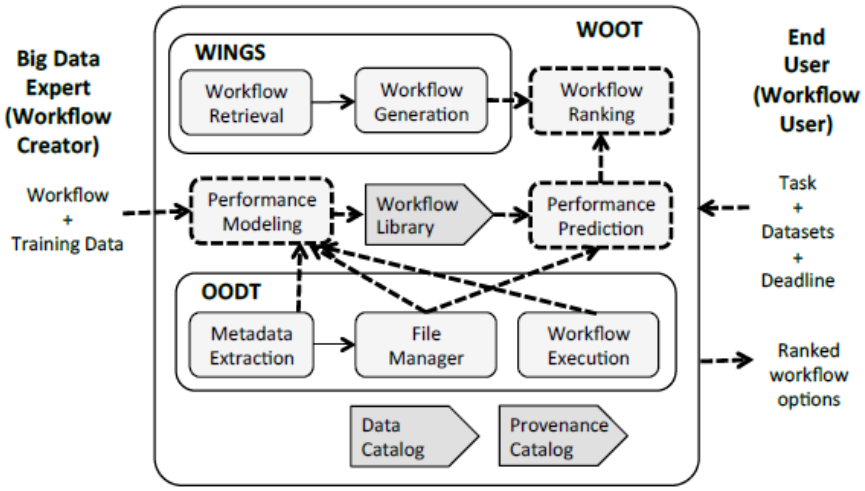


Figure 2: A high-level view of the system, with the new functionality of WOOT highlighted with dashed lines. During the learning phase, the system uses training datasets (of different sizes and characteristics) to run workflows in OODT and extract metadata to be used as features to train performance models of the workflows. When the system is in use and the user provides a user request, Wings retrieves relevant workflows (all workflows that include a topic modeling step), specializes them into instantiated workflows, ranks them according to their performance, and offers them to the user as options. The workflows selected by the user are submitted for OODT for execution.

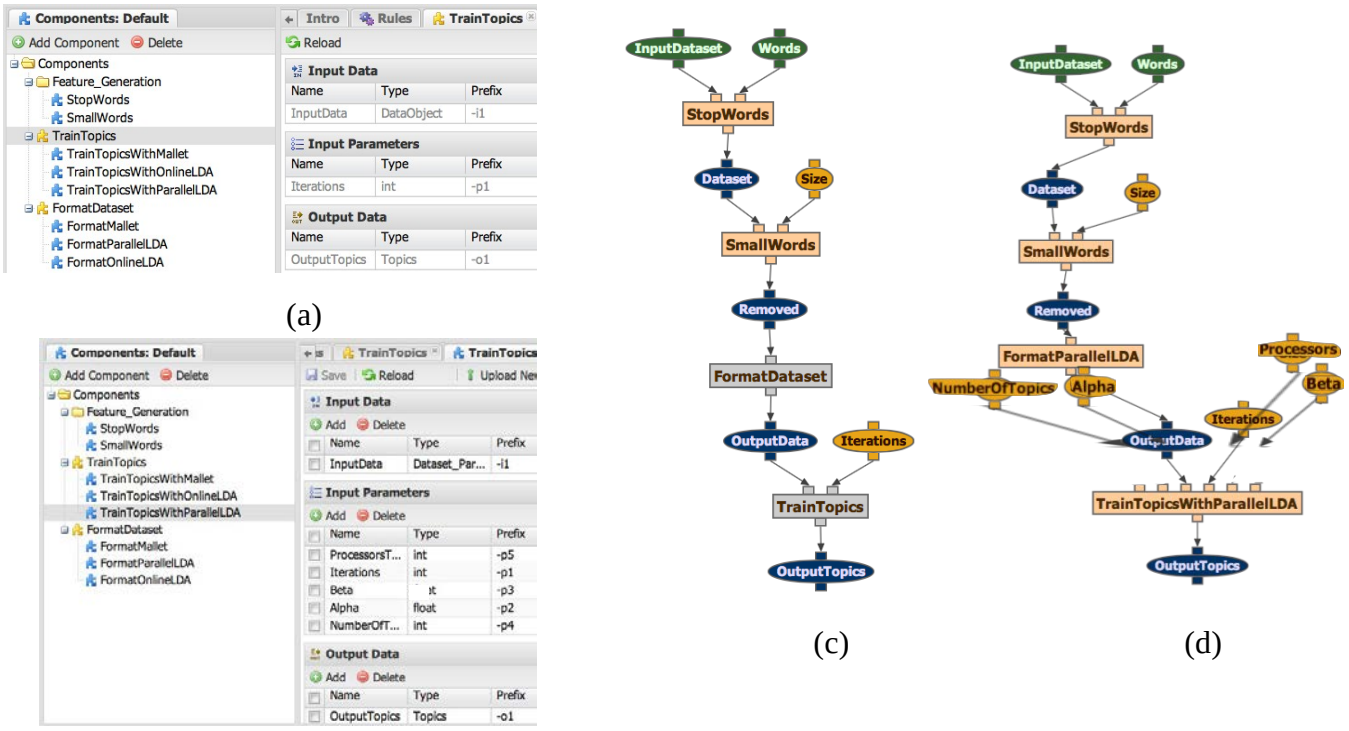


Figure 3: Wings allows workflow designers to organize workflow components into hierarchies, and use component classes as abstract steps. A component class is shown in (a), where TrainTopics represents a class of workflow components that have one input dataset, an Iterations parameter, and an OutputTopics dataset. A workflow component under that class is shown in (b), inheriting those three characteristics and having additional parameters such as Processors, Alpha, Beta, and NumberOfTopics. Workflow designers can create abstract workflow templates with the component class TrainTopics as a step, as shown in (c). Wings automatically generates specializations of that template including the workflow in (d) and other workflows shown in Fig 1.

5.1 Creating Workflows

Workflow creators are big data experts that have experience using the different algorithms available with different datasets and parameters that impact their performance. When creating a new workflow, they are asked to specify some sample datasets of different sizes and characteristics, as well as key parameter values. These datasets will be used to form a comparative model that can be leveraged to predict characteristics of a given workflow, as we describe next. Clearly the data sets associated with the workflow should also be similar in nature, such that a more accurate comparison can be established.

Workflow creators are offered the ability to create abstract workflow templates that include steps that do not refer to specific algorithms or implementations, but instead refer to *classes* of algorithms that carry out a similar task.

Figure 3(a) shows an example of such a class hierarchy for topic modeling workflows that an expert would create. The class TrainTopics includes three different algorithms represented as subclasses: Mallet-LDA, Online LDA, and Parallel LDA. All three algorithms take an input dataset, a parameter indicating the number of iterations, and output the topic models. These common properties are represented in the class TrainTopics and inherited to each of the three subclasses. Since each algorithm has its own parameters, those are represented in their subclass as shown in Figure 3(b).

Wings allows a workflow creator to use component classes as steps in specifying a workflow template. Figure 3(c) shows an example, where one of the steps is TrainTopics, and another step is also an abstract class (FormatDataset). These abstract workflows[1] are presented to the end users, and are automatically specialized to the algorithms available as we describe in Section 5.4 below. An example of a specialized workflow is shown in Figure 3(d), others include those shown in Figure 1(a)(b)(c).

To train a performance model for the topic modeling workflows discussed here, we used the datasets in http://csmining.org/index.php/r52-and-r8-of-reuters-21578.html, which are public datasets of document collections that contain news items widely used in the natural language processing and machine learning communities. The dataset sizes are as follows: R8_train: 3.2MB, R8_test: 1.1MB, R52_train: 4.1MB, R52_test: 1.5MB. We then created additional datasets by selecting random subsets of those document collections.

As for parameters, we assume that users provide a selected set of parameter values to try. For example, in the case of MALLET LDA the parameter that sets the number of iterations is recommended to be set between 1000 to 2000, so the user can choose a few sample values from that range.

5.2 Learning Workflow Performance Models

Using the datasets and parameter values provided by the workflow creator, the system then runs the workflow with all the possible combinations. OODT records all the performance information for each workflow run. It also extracts and records metadata about the input datasets. The WOOT Performance metadata of datasets used as well as parameter settings. These form the features (or variables) for learning the performance models. For example:

```
OnlineLDA-Workflow
input1 R8_test size 1.1MB numLines 100,000
num-topics 10
num-iterations 1000
optimize-interval 10
optimize-burn-in 20
output-state-interval 0
runtime 160
```

Note that the second line has the ID of the input dataset followed by its metadata represented as key value pairs. OODT can extract this metadata and annotate it on the provenance record of the workflow execution.

Given this information, the system learns a performance model by doing a linear regression on the workflow execution data collected from OODT.

Figure 4 shows a performance model for the Online LDA workflow. The features that we used included the size of the input file, the number of lines of the input file, and the values of all the parameters set for the specific run. Shown in the figure are the size and the iterations parameter, the latter affecting runtime more dramatically. Additional metadata properties to train the performance model can be extracted through OODT using Apache Tika [Mattmann and Zitting 2011], such as the number of distinct words, the language of the file, the file format (html, plain text, etc).

The performance modeling function is very flexible and extensible and operates in the following way: (1) utilizes extension points for incorporating various runtime estimation algorithms, (2) provides run time estimation, among other calculations, via a RESTful web-application tier, and (3) has network connectors to multiple OODT components, such as the File Manager and Workflow Manager, to enable a more comprehensive approach to workflow runtime modeling. Wings will invoke this function when trying to rank candidate workflows for the user, as will be described in Section 5.5.

5.3 User Request

A workflow user is an end user who has a particular data analytic task at hand, but does not have the analytic expertise of the workflow creator who creates workflows as discussed in Section 5.1. They would be presented with a collection of abstract workflows for different tasks. For example, for the task of topic detection they could be shown the workflow in Figure 3(c). Other tasks could include document clustering and document classification; we discuss such collections of text analytics tasks in [Hauder et al 2011a]. It is easy for novices to select workflows based on the tasks that they performed, as we showed in [Hauder et al 2011b].

The user can specify the dataset that they want to analyze, and a time bound in minutes. The user can leave the parameters unspecified. The parameters are automatically selected by the system, as we describe next.

[1] In other research, abstract workflow refers to a workflow with no resources specified to the workflow tasks. Here, abstract workflow refers to workflows with no particular algorithms specified, therefore introducing an additional abstraction layer.

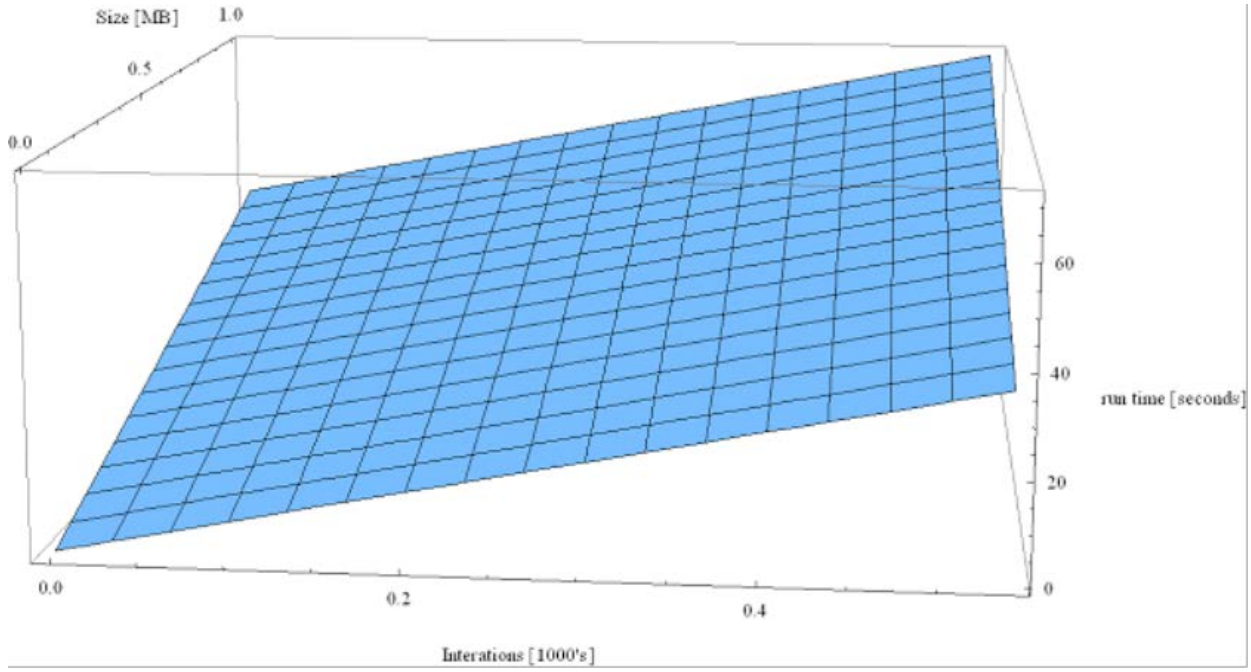


Figure 4: A partial plot of the performance model for the Online LDA workflow shown in Figure 1(b).

5.4 Automatic Generation of Candidate Workflows

Given an abstract workflow template and input datasets, Wings can automatically specialize the workflows by performing a search through the space of possible workflows whose steps are specific algorithms that are consistent with that abstract workflow. An overview of the algorithm is given in [Gil et al 2011a], a detailed description can be found in [Gil et al 2011b]. We briefly summarize it here. First, Wings specializes the workflow steps by replacing component classes with the possible subclasses, each generating a branch in the search for candidate workflows. Then, Wings assigning values to all unspecified parameters. Any workflow that is fully elaborated through this search can be submitted to OODT for execution.

5.5 Ranking Candidate Workflows

For each candidate workflow, the WOOT Workflow Ranker requests an estimate of the workflow runtime from the WOOT Performance Modeler described in Section 5.2. The request would be given as follows:

```
OnlineLDA-Workflow
input1  R52_test
num-topics 10
num-iterations 1000
optimize-interval 10
optimize-burn-in 20
output-state-interval 0
```

The WOOT Performance Modeler would return:

```
OnlineLDA-Workflow
input1  R52_test    size  1.5MB    numLines 135,000
num-topics 10
num-iterations 1000
optimize-interval 10
```

```
optimize-burn-in 20
output-state-interval 0
runtime-estimate 185
```

Note that the system has requested from OODT the metadata needed, including the size and number of lines in the input file. Those metadata properties and values, together with the parameter values, are used by the WOOT Performance Modeler to generate a runtime estimate, returned in the last line.

5.6 Presenting Users with Workflow Options

Figure 5 illustrates how the system presents workflow options to the end user. Each line represents a possible workflow candidate, and can be selected for submission to OODT. The workflow options are not shown as a dataflow graph as workflows are typically shown, since they all share the same dataflow graph represented by the abstract workflow template. To highlight the differences between the alternative workflows, we show for each workflow candidate: 1) the particular algorithms that would be used for each step, 2) the input data, 3) the parameter values, and 4) the estimated runtime. The workflow candidates can be sorted according to runtime. The user then selects one or more workflows for execution, which would be submitted to OODT.

The workflow candidates shown in Figure 5 correspond to the 3 LDA algorithms discussed in Section 2, and the system is suggesting the best parameter values available. Note that our datasets are relatively small, but if they were larger they would have dramatically different runtime estimates. This is also the reason why the parallel LDA algorithm has the worst performance, since the parallelization creates overhead and typically is not an efficient way to process a small dataset.

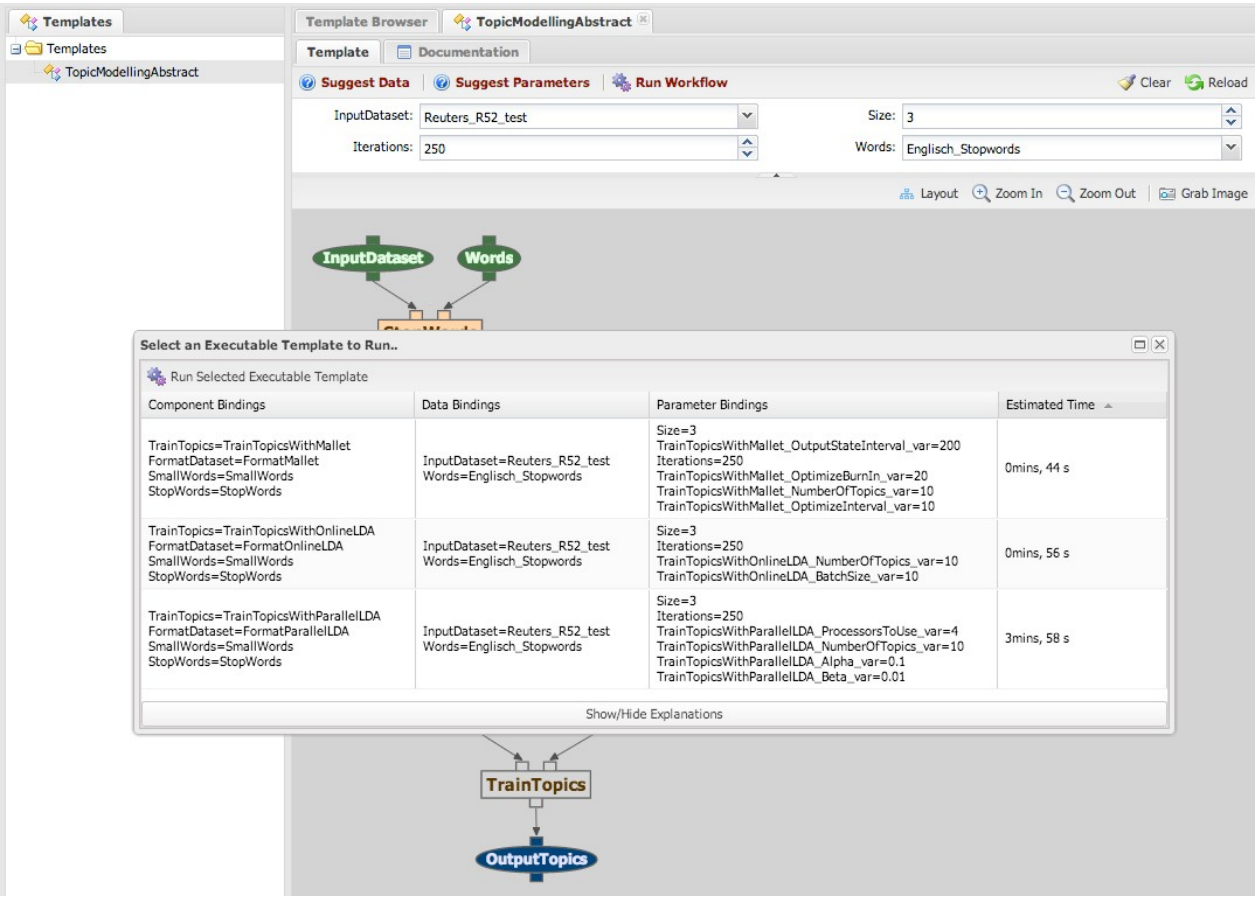


Figure 5: Once the end user selects the abstract workflow template shown in Figure 3(c), the system presents suggests different alternative workflows that use different algorithms and have different runtimes. The user can select the faster one, or select an algorithm that they recognize and prefer. The system could show other estimates in addition to runtime, such as expected accuracy of the answer, general reliability, and other criteria relevant to an end user.

6. DISCUSSION

Current limitations of our system are topics for future work. An important capability that we could provide is to compare workflows along dimensions other than runtime. These could include domain-specific comparative assessments across workflow options. For example, an extension that would be easy to do in our framework is to allow the WOOT Workflow Ranker to estimate the quality of the solutions. The creator of each workflow would be asked to provide an assessment of the quality of the workflow output as a function of the parameters of the workflow. This could be written as a set of rules for the workflow, and they have to be designed in such a way that each set of parameter values leads to a single quality assessment.

For example, for the workflow WF-LDA-MALLET in Figure 1(a) the following rules could be provided:

```
If the number of iterations is less than 1000,
    then the result quality is LOW.

If the number of iterations is more than 1500,
    then the result quality is HIGH.

If the number of iterations is more than 1000
    and less than 1500, then the result
    quality is MEDIUM.
```

Therefore, each workflow instance would have an associated quality estimate and an associated runtime estimate, giving users the ability to explore performance/quality tradeoffs. Such rules for the alternative algorithms are highly domain-specific, but they represent knowledge that is very familiar to big data experts who have run the algorithms many times themselves and assessed the quality of the results.

We are also exploring alternative selection algorithms based on Bayesian statistics [Winkler 2003]. In this extension, a workflow designer would also provide an assessment of workflow quality e.g., as described (HIGH,MEDIUM,LOW) given a set of workflow inputs (e.g., such as data size). These relations form a mapping between observation of a particular workflow input value, or constraint, and quality of workflow. The combination of the set of these observed values and the workflow designers assessed workflow quality given those values create a conditional probability distribution over the space of features, such as:

```
P(HIGH|data size<1Gb) = 0.75
P(MEDIUM|data size<1Gb) = 0.2
P(LOW|data size<1Gb) = .05
..
```

Our future work would take in the above conditional probability distribution as workflow designer input, then execute a Bayesian inference/selection algorithm to combine the conditional probably information into an overall probably for a workflow's quality, given its associated metadata, input, and then this probably would yield a ranking for the set of workflows being assessed.

Another dimension of improvement for our system is the interaction with the user, which could be made more sophisticated. For example, if the user gives a time bound that is not possible with their data and parameters selected, then the system could explain how far is the time bound from what the user needs and suggest choosing a very different abstract workflow for a similar task but perhaps not as thorough. Another possibility would be to show the end user k qualitatively different workflows that used very different algorithms and had very different performance estimates. All of them could be submitted for execution, and the results shown to the user as alternative solutions to their task.

We could also create more accurate and refined performance models building on prior work [Miu and Missier 2012; Montagnat et al 2010; Vahi et al 2012; Furlani et al 2013; Carrington et al 2005; Hutter et al 2012]. More refined performance models could take into account dynamic factors such as network latency, queue wait time, and availability of resources required to run a given workflow. Conversely, those frameworks could use our approach to include additional features for the performance estimates, based on the metadata properties that we use.

Another extension would be to improve the system performance and confidence as more workflows are run. Once a workflow has been executed, WOOT has the predicted runtime and the actual runtime. Clearly it could use the actual runtime to improve its performance model for that workflow. But in addition, it could use both values to create a measure of its confidence on the runtime estimates for the workflow and present those to the user as an indication of uncertainty.

7. CONCLUSIONS

In order to enable end users of big data systems to find workflows that suit their needs given a task and a deadline, we presented an approach and implemented system that automatically generates alternative workflow candidates and presenting them to the user in a rank order according to performance estimates. The system makes these estimates based on workflow performance models, and uses semantic technologies to reason about workflow options and their quality. We implemented this approach in the WOOT system, which combines and extends capabilities from the Wings semantic workflow system and the Apache OODT Object Oriented Data Technology and workflow execution system.

8. ACKNOWLEDGMENTS

We thank the WINGS and Apache OODT teams for their support of this work. We gratefully acknowledge support from the Defense Advanced Research Projects Agency (DARPA) with award FA8750-13-C-0016, and from the Air Force Office of Scientific Research (AFOSR) with award FA9550-11-1-0104.

9.REFERENCES

[1] Blei, D., Ng, A., and M. Jordan. "Latent Dirichlet Allocation." Journal of Machine Learning Research, 3, pp 993–1022, January 2003.

[2] Carrington, L. C. et al. "How Well Can Simple Metrics Represent the Performance of HPC Applications?", Proceedings of the 2005 ACM/IEEE conference on Supercomputing, 2005.

[3] De Roure, D; Goble, C.;Stevens, R. "The design and realizations of the myExperiment Virtual Research Environment for social sharing of workflows". Future Generation Computer Systems, 25 (561-567), 2009.

[4] Furlani, T. R., Jones, M. D., Gallo, S. M., Bruno, A. E., Lu, C., Ghadersohi, A., Gentner, R. J., Patra, A., DeLeon, R. L., von Laszewski, G., Wang, F., and A. Zimmerman. Performance metrics and auditing framework using application kernels for high-performance computer systems. Concurrency and Computation: Practice and Experience, 25(7), 2013.

[5] Gil, Y., Groth, P., Ratnakar, V., and C. Fritz. "Expressive Reusable Workflow Templates." Proceedings of the IEEE e- Science Conference, Oxford, UK, pages 244–351. 2009.

[6] Gil, Y.; Deelman, E.; Ellisman, M. H.; Fahringer, T.; Fox, G.; Gannon, D.; Goble, C. A.; Livny, M.; Moreau, L.; and Myers, J. "Examining the Challenges of Scientific Workflows." IEEE Computer, 40(12), 2007.

[7] Gil, Y.; Ratnakar, V.; Kim, J.; Gonzalez-Calero, P. A.; Groth, P.; Moody, J.; and Deelman, E. "Wings: Intelligent Workflow-Based Design of Computational Experiments." IEEE Intelligent Systems, 26(1). 2011.

[8] Gil, Y.; Gonzalez-Calero, P. A.; Kim, J.; Moody, J.; and Ratnakar, V. "A Semantic Framework for Automatic Generation of Computational Workflows Using Distributed Data and Component Catalogs." Journal of Experimental and Theoretical Artificial Intelligence, 23(4), 2011.

[9] Hauder, M., Gil, Y. and Liu, Y. "A Framework for Efficient Text Analytics through Automatic Configuration and Customization of Scientific Workflows". Proceedings of the Seventh IEEE International Conference on e-Science, Stockholm, Sweden, December 5-8, 2011.

[10] Hauder, M.; Gil, Y.; Sethi, R.; Liu, Y.; and Jo, H. "Making Data Analysis Expertise Broadly Accessible through Workflows." Proceedings of the Sixth Workshop on Workflows in Support of Large-Scale Science (WORKS'11), held in conjunction with SC 2011, Seattle, WA, 2011.

[11] Hoffman, M., Blei, D., and F. Bach. "Online Learning for Latent Dirichlet Allocation." NIPS, 2010.

[12] Hutter, F., Xu, L., Hoos, H. H., and K. Leyton-Brown. "Algorithm Runtime Prediction: The State of the Art". Available from arXiv:1211.0906.

[13] Langford, J. Vowpal Wabbit. https://github.com/JohnLangford/vowpal_wabbit/ 2011.

[14] Mattmann, C. A., et al. "A reusable process control system framework for the orbiting carbon observatory and NPP Sounder PEATE missions." Third IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT), 2009.

[15] Mattmann, C. A., and J. Zitting. "Tika in Action." Manning Publications, 2011.

[16] McCallum, A. K. "MALLET: A Machine Learning for Language Toolkit." http://mallet.cs.umass.edu. 2002.

[17] Miu, T. and P. Missier. Predicting the Execution Time of Workflow Activities Based on Their Input Features, Proceedings of the Seventh Workshop on Workflows in Support of Large-Scale Science (WORKS'12), held in conjunction with SC 2012.

[18] Montagnat, J., Glatard, T., Reimert, D., Maheshwari, K., Caron, E., and F. Desprez. "Workflow-based comparison of two Distributed Computing Infrastructures." Proceedings of the Fifth Workshop on Workflows in Support of Large-Scale Science (WORKS'10), New Orleans, LA, 2010.

[19] Řehůřek, R. gensim. http://radimrehurek.com/gensim/. 2009.

[20] Vahi, K., Harvey, I., Samak, T., Gunter, D. K., Evans, K., Rogers, D., Taylor, I., Goode, M., Silva, F., Al-Shakarchi, E., Mehta, G., Jones, A. and E. Deelman. "A General Approach to Real-Time Workflow Monitoring." Proceedings of the Seventh Workshop on Workflows in Support of Large-Scale Science (WORKS'12), 2012.

[21] Wang, Y., Bai, H., Stanton, M., Chen, W., and E. Y. Chang. "PLDA: Parallel Latent Dirichlet Allocation for Large-Scale Applications." AAIM, 2009.

[22] Winkler, R. L. "An Introduction to Bayesian Inference and Decision. Probabilistic Press, 2003.

8.0 LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

ASF: Apache Software Foundation
CAS: Catalog and Archive System
CIDR: Classless Inter-Domain Routing
ETL: Extract-Transform-Load
Gephi: Open Graph Visualization Platform
GISR: Global Intelligence, Surveillance, and Reconnaissance
GMTI: Ground Moving Target Indicator
HDFS: Hadoop Distributed File System
JSON: JavaScript Object Notation
LDA: Latent Dirichlet Allocation
NTC: National Training Center
OODT: Object Oriented Data Technology
OpsUI: Open Source User Interface
PGE: Product Generation Engine
REST: Representational State Transfer
Solr: Open source enterprise search platform
WINGS: Workflow Instance Generation and Selection
XNET: XDATA Network